TUGAS AKHIR – TE 141599

Perancangan Kontroler *Adaptive* PID untuk *Electromagnetic Anti-Lock Braking System* (ABS) pada Kendaraan Listrik

Muhammad Fadli Ilmi
NRP. 2211 100 193

Dosen Pembimbing
Ir. Josaphat Pramudijanto, M.Eng.
Andri Ashfahani, ST., MT.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT – **TE 141599**

Design of Adaptive PID Controller for Electro-Magnetic Anti-Lock Braking System (ABS) on Electric Vehicle

Muhammad Fadli Ilmi
NRP. 2211 100 193

Supervisor
Ir. Josaphat Pramudijanto, M.Eng.
Andri Ashfahani, ST., MT.

DEPARTEMENT OF ELECTRICAL ENGINEERING
Faculty of Industrial Technology
Sepuluh Nopember Insitute of Technology
Surabaya 2016

PERANCANGAN KONTROLER *ADAPTIVE* PID UNTUK *ELECTROMAGNETIC ANTI-LOCK BRAKING SYSTEM* (ABS) PADA KENDARAAN LISTRIK

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Sistem Pengaturan
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Josaphat Pramudijanto, M.Eng.	**Andri Ashfahani, ST., MT.**
NIP. 196210051990031003	NIP.2200201405003

SURABAYA
JANAURI, 2016

Perancangan Kontroler *Adaptive* PID untuk *Electromagnetic Anti-Lock Braking System* (ABS) pada Kendaraan Listrik

Muhammad Fadli Ilmi
2211 100 193

Dosen Pembimbing I : Ir. Josaphat Pramudijanto, M.Eng.
Dosen Pembimbing II : Andri Ashfahani, ST., MT

ABSTRAK

Anti-Lock Braking System (ABS) digunakan pada *automobile* untuk meminimalkan jarak pemberhentian dan mencegah roda terkunci pada saat pengereman mendadak. Roda sering terkunci pada saat pengereman mendadak pada jalan yang koefisien gesekannya kecil. Selama roda terkunci, kendaraan akan kehilangan kontrol *steering* dan gaya geseknya berkurang secara cepat sehingga kendaraan sulit untuk berhenti. Dalam kondisi normal, kecepatan kendaraan hampir sama dengan kecepatan roda, namun pada keadaan roda terkunci dan mengalami *slip*, perbedaan kecepatan kendaraan dan kecepatan roda sangat jauh. Secara umum, ABS menawarkan keamanan pada kendaraan dengan membatasi *slip* roda longitudinal saat pengereman dengan kondisi *slip* tinggi. Pada tugas akhir ini digunakan metode *Gain Scheduling* untuk mengontrol ABS. Konsep kontroler digunakan untuk mengatur dan menyesuaikan *error slip* dan *error output* dari kendaraan pada saat terjadi pengereman dengan kondisi jalan tertentu, Hal ini digunakan untuk mempertahankan *slip* dalam spesifikasi yang diberikan, yaitu 20 persen. Masukkan *plant* ABS adalah slip yang diharapkan dan keluarannya adalah slip dari *plant*. Kontroler PID pada saat simulasi dan implementasi dapat mengikuti *set point* yaitu sebesar 0,2 pada setiap kondisi jalan saat pengereman terjadi.

Kata Kunci : *ABS, PID,Gain Scheduling*

Design of Adaptive PID Controller for Electro-Magnetic Anti-Lock Braking System (ABS) on Electric Vehicle

Muhammad Fadli Ilmi
2211 100 193

Supervisor I : Ir. Josaphat Pramudijanto, M.Eng.
Supervisor II : Andri Ashfahani, ST., MT.

ABSTRACT

Anti-lock Braking System (ABS) is used on the automobile to minimizing the distance stops and prevents the wheels locked when braking suddenly. The wheels are often locked at the time of sudden braking on the road which coefficient traction is small. As long as the wheels are locked, the vehicle will lose control of the steering and the traction force reduced quickly so that the vehicle is hard to stop. Under the normal conditions, the speed of the vehicle is almost the same as the speed of the wheels, but on the State of the locked wheels and slip, the difference in the speed of the vehicle and the speed of the wheels is very far away. In General, the ABS offers safety on the vehicle by limiting longitudinal wheel slip when braking with high slip condition. In this final project, a method of Gain Scheduling for controlling the ABS controller concept is used to set up and customize error slips and errors output of vehicles in when braking with certain road conditions, it is used to maintain a slip in the specifications provided, which is 20%. Input of plant ABS is desired slip and output of plant ABS is slip from plant PID controller at the time of the simulation and implementation can follow the set point which is 0.2 on any road conditions when braking occurs.

.

Keywords **:** *ABS, PID, Gain Scheduling*

KATA PENGANTAR

Alhamdullilah, puji syukur penulis panjatkan kehadirat Allah SWT karena atas rahmat dan karunia-Nya penulis dapat menyelesaikan penulisan buku tugas akhir dengan judul **"PERANCANGAN KONTROLER *ADAPTIVE* PID UNTUK *ELECTROMAGNETIC ANTI-LOCK BRAKING SYSTEM* (ABS) PADA KENDARAAN LISTRIK"**. Tugas akhir merupakan salah satu syarat yang harus dipenuhi untuk menyelesaikan program studi Strata-1 pada Jurusan Teknik Elektro Fakultas Teknologi Industri Institut Teknologi Sepuluh Nopember Surabaya.

Penulis menyadari bahwa dalam penulisan tugas akhir ini banyak mengalami kendala, namun berkat bantuan, bimbingan, dan kerja sama dari berbagai pihak sehingga kendala-kendala tersebut dapat diatasi. Untuk itu pada kesempatan ini penulis menyampaikan banyak terimakasih dan penghargaan setingi-tingginya kepada :

1. Kedua orang tua, Ayahanda Wawan Hermawan dan Ibunda Cicah Rodisah yang selalu memberikan dukungan, semangat, dan doa kepada penulis.
2. Bapak Josaphat dan Bapak Andri selaku Dosen Pembimbing atas segala bantuan, perhatian, dan arahan selama pengerjaan tugas akhir ini.
3. Bapak Rusdi selaku Koordinator Bidang Studi Sistem Pengaturan Jurusan Teknik Elektro ITS.
4. Bapak Ardiono selaku Ketua Jurusan Teknik Elektro ITS.
5. Rekan-rekan e51 khususnya bidang studi Sistem Pengaturan.
6. Teman-teman seperjuangan, dan teman satu kontrakan.

Penulis berharap tugas akhir ini dapat bermanfaat bagi yang membutuhkannya.

Surabaya, 20 Januari 2016

Penulis

DAFTAR ISI

TABLE OF CONTENT

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Pengereman merupakan salah satu bagian penting dari sistem kendaraan. Sistem pengereman yang baik akan memberikan rasa aman bagi pengendara, terutama untuk kondisi jalan yang berpasir maupun basah. Pada saat pengereman mendadak roda dapat mengunci arahnya sehingga mobil tidak terkendali (*understeering)* dan menyebabkan kendaraan tegelincir. Salah satu sistem yang dapat mengatasi masalah pengereman tersebut adalah *Anti-Lock Braking System* (ABS). *Anti-Lock Braking System* (ABS) adalah sistem yang terkontrol secara otomatis untuk mencegah rem terkunci. Beberapa keuntungan dari ABS adalah untuk menjamin kenyamanan dan kestabilan pengereman pada semua kondisi jalan, meningkatkan pengendalian kemudi kendaraan (*steering ability*), mengurangi jarak pengereman, mengurangi gaya sentrifugal saat melakukan pengereman mendadak yang dapat membuat mobil susah dikendalikan.

Rem elektromagnetik adalah sistem pengereman yang menggunakan gaya elektromagnetik untuk memperlambat gerakan poros roda kendaraan. Rem elektromagnetik terbuat dari sebuah piringan dengan bahan logam *non ferromagnetic* yang terpasang pada poros yang berputar. Piringan tersebut diapit oleh sisi stator yang berupa lilitan elektromangetik yang dapat membangkitkan medan magnet ketika dialiri arus listrik. Ketika arus listrik melewati lilitan tersebut akan menimbulkan medan magnet pada lilitan dan piringan yang memotong medan magnet akan menimbulkan arus *eddy* pada piringan itu sendiri. Arus *eddy* ini akan menimbulkan medan magnet yang arahnya berlawanan dengan medan manget sebelumnya.

Kontroler tradisional seperti PID tidak dapat memenuhi kebutuhan dari ABS, sebab kontroler PID tidak dapat menentukan parameter dinamik dan tidak dapat mengurangi efek gangguan pada sistem. Kontrol adaptif merupakan metode kontrol cerdas dengan *adjustable parameter* dan mekanisme untuk mengatur parameter atau dalam penegertian umumnya adalah mengubah karakteristik untuk menyesuaikan diri terhadap keadaan yang baru atau tidak diketahui. Kontrol adaptif dapat mengatasi *plant* yang nonlinier seperti pada ABS. Namun kontrol adaptif sulit mengatasi *error steady state*, sehingga dibutuhkan kolaborasi dua metode kontrol tersebut.

1.2 Perumusan Masalah

Adapun rumusan masalah dari pelaksanaan tugas akhir ini, antara lain ABS belum mampu meminimalkan *error slip* dari motor yang melaju pada lintasan dengan kondisi lingkungan yang berbeda-beda. Kemudian apakah PID adaptif mampu mengatur ABS sehingga *error slip* mencapai nilai yang diinginkan sebesar 10-30%.

1.3 Batasan Masalah

Pada Tugas Akhir ini, pembahasan masalah dibatasi pada *interface* yang digunakan berupa Arduino UNO R3, hasil identifikasi pendekatan model sistem berupa *Auto Reggressive*, data kecepatan pada Matlab berupa *Transfer function* pada simulink Matlab, penentuan koefisien gesek berdasarkan pada referensi yang sudah ada dan pada kondisi jalan datar. *Plant* ABS dikerjakan oleh satu tim, sehingga setiap orang dibagi tugasnya, dalam hal ini penulis lebih banyak mengerjakan perancangan pada *hardware. Plant* ABS yang digunakan merupakan hasil modifikasi dari *plant* ABS tim terdahulu yang dikerjakan oleh Rian Lukito, Muhammad Fasih Mubarrok, Benny Adijaya, dan Muhammad Fadli Ilmi.

1.4 Tujuan Penelitian

Tujuan dari pelaksanaan tugas akhir ini, antara lain untuk meminimalkan *error slip*, mengatur kecepatan roda kendaraan yang tepat sesuai dengan kondisi jalan dan gesekan pada roda dan untuk membantu meningkatkan kemampuan meminimalisasi jarak pemberhentian kendaraan saat terjadi pengereman mendadak.

Manfaat dari pelaksanaan tugas akhir ini, antara lain sebagai salah satu acuan pembelajaran terhadap cara mendesain dan mengimplementasikan kontroler *Adaptive* PID untuk sistem ABS.

1.5 Sistematika Penulisan

Sistematika pembahasan dalam penyusunan buku tugas akhir ini, terdiri dari lima bab yaitu pendahuluan, teori dasar, perancangan sistem, pengujian dan analisa serta kesimpulan dan saran. dijelaskan sebagai berikut:

BAB I: PENDAHULUAN

Menguraikan latar belakang, perumusan masalah, batasan masalah, tujuan, metedologi, sistematika penulisan dan relevansi yang digunakan dalam pembuatan tugas akhir.

BAB 2: TEORI DASAR

Berisikan tentang pembahasan mengenai Arduino UNO, kontroler PID, model dinamika *plant*.

BAB 3: PERANCANGAN SISTEM DAN KONTROLER

Membahas tentang perencanaan dan pembuatan sistem yang dibangun, seperti perancangan kontroler, identifikasi *plant*, validasi *plant*.

BAB 4: PENGUJIAN DAN ANALISA

Menjelaskan tentang hasil pengujian sistem yang sudah dibuat, dengan analisa mengenai hasil yang diperoleh.

BAB 5: KESIMPULAN DAN SARAN

Merupakan bab penjelasan tentang pengambilan kesimpulan dari analisa dan hasil pengujian yang telah diperoleh.

1.6 Relevansi

Diharapkan dari tujuan pengerjaan tugas akhir ini dapat memberikan manfaat dan menjadi referensi dalam perancangan simulator *plant* suatu proses industri yang dapat digunakan dalam perkuliahan. Pengaplikasian ilmu kontrol dalam perancangan kontroler dan implementasi komunikasi data dalam industri, khususnya kontroler PID *adaptive*.

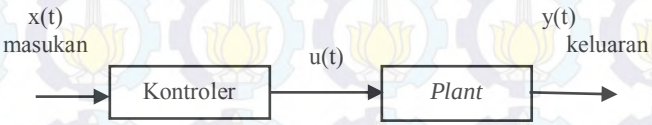
BAB 2
TEORI DASAR

Pada bab ini, berisi tentang dasar teori yang digunakan sebagai landasan dalam mengerjakan tugas akhir. Teori tersebut meliputi dasar sistem kontrol, elektromagnetik *Anti-Lock Braking System*, kontrol yang digunakan, identifikasi sistem.

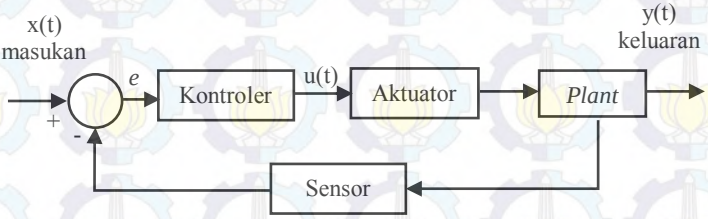
2.1 Dasar Sistem Kontrol Proses [1]

Sistem kontrol adalah proses pengaturan terhadap satu atau beberapa besaran (parameter, variabel), sehingga berada pada suatu harga tertentu. Tujuan dari sistem kontrol adalah untuk mendapatkan titik kerja yang optimal pada suatu sistem yang dirancang.

Berdasarkan aksi kontrolnya sistem kontrol dapat dibagi menjadi dua bagian, yaitu sistem kontrol *open loop* dan sistem kontrol *closed loop*. Sistem kontrol *open loop* adalah suatu sistem kontrol yang aksi kontrolnya tidak dipengaruhi oleh *output* dari sistem, sedangkan sistem kontrol *closed loop* adalah sistem kontrol yang aksi kontrolnya bergantung pada *output* dari sitem.



Gambar 2.1 Diagram Blok Sistem Kontrol *Open loop*



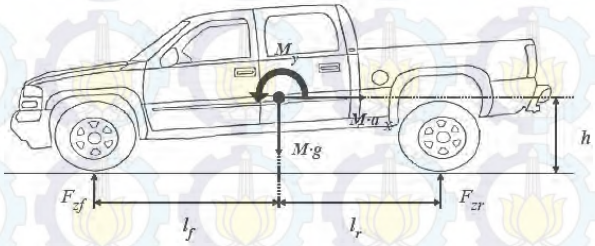
Gambar 2.2 Diagram Blok Sistem Kontrol *Closed Loop*

Seperti yang ditunjukkan pada Gambar 2.1 dan Gambar 2.2, perbedaan antara *open loop* dan *closed loop* adalah ada atau tidaknya umpan balik *output* untuk dibandingkan kembali dengan *input*.

Pada Gambar 2.2, bagian kontroler memiliki *summing point* dengan tanda (+/-). Di titik inilah langkah membandingkan dilakukan dengan menggunakan besaran set point dengan sinyal hasil pengukuran. Hasilnya adalah berupa sinyal kesalahan.

2.2 Dinamika Kendaraan [2]

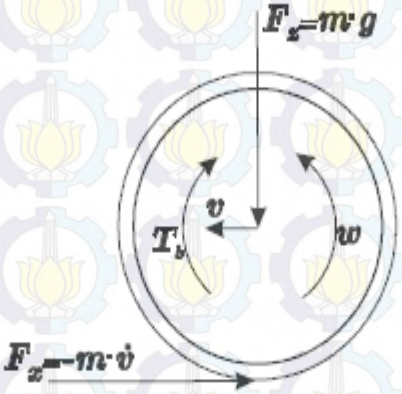
Model *longitudinal* dari kendaraan, seperti pada Gambar 2.3, mempertimbangkan beberapa batasan, yakni, keempat roda tetap menempel pada permukaan jalan saat pengereman dan moment yaw serta moment *roll* mobil bernilai nol. Selain itu, gaya aerodinamis diasumsikan sangat kecil dan diabaikan.



Gambar 2.3 Model *Longitudinal* Kendaraan

Salah satu gaya yang bekerja pada kendaraan dari keadaan awal bergerak hingga berhenti adalah gaya pengereman yang berlawanan arah dengan arah laju kendaraan. Pada gaya pengereman bekerja maka dinamika kendaraan akan berubah pada 3 sumbu *axis*. Pada bahasan kali ini, dinamika kendaraan yang akan dibahas adalah pada sumbu *axis longitudinal*. Pada *axis longitudinal*, efek pengereman akan mengakibatkan perubahan pada dua parameter, yaitu kecepatan *longitudinal* kendaraan dan kecepatan putar roda. Kecepatan putar dan kecepatan *longitudinal* akan mengalami *slip* pada saat pengereman. Nilai *slip* ini perlu dikendalikan pada nilai 0%-20% agar kendaraan dapat dikemudikan pada saat pengereman mendadak. *Slip* terjadi pada masing-masing roda pada sebuah kendaraan yang dipengaruhi oleh koefisien gesek. Dinamika gaya yang terjadi saat pengereman pada sebuah roda

adalah gaya-gaya yang bekerja pada sebuah roda. Saat pengereman gaya yang bekerja adalah *braking force*, gaya normal kendaraan dan gaya gesek jalan. Pengaruh gaya yang terjadi dapat dilihat pada Gambar 2.4.



Gambar 2.4 Gaya-Gaya yang terjadi pada Roda

Gaya-gaya yang terjadi diakibatkan oleh dinamika gerak suatu mobil. Berikut persamaan gerak pada mobil:

$$m\dot{v} = -F_x \tag{2.1}$$

$$J\dot{\omega} = rF_x - T_b sign(\omega) \tag{2.2}$$

Di mana:

m= massa
v = kecepatan horizontal saat mobil bergerak
ω= kecepatan angular roda
F_x= gaya gesek roda
F_z= gaya vertikal
r = jari-jari pada roda
J = inersia pada roda

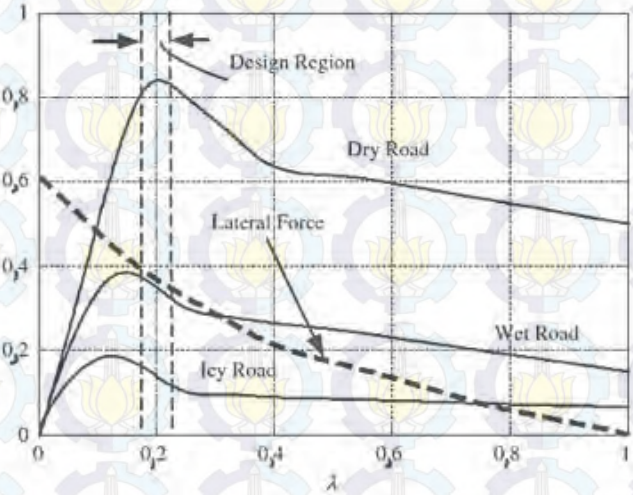
Gaya gesek ban dapat dilihat pada persamaan berikut:

$$F_x = -F_z \,.\, \mu(\lambda, \mu_r, \alpha) \tag{2.3}$$

Dimana $\mu_r$adalah koefisisen gesek antara jalan dan roda, λ adalah *slip* pada roda, dan α adalah sudut *slip* pada roda.

2.3 *Anti-Lock Braking System* [3]

Anti-Lock Braking System merupakan sistem pengereman pada mobil agar tidak terjadi penguncian roda saat pengereman mendadak. Kemampuan ABS tersebut berasal dari hubungan antara koefisien gesek jalan (μ_r) dengan *slip* rasio roda (λ). Koefisien gesek jalan tergantung dari beberapa faktor, yakni kondisi permukaan jalan (kering atau basah), sudut roda yang menempel dipermukaan, jenis roda, kecepatan kendaraan, dan *slip* rasio antara roda dengan permukaan jalan.



Gambar 2.5 Koefisien Gesek vs *Slip* pada Beberapa Kondisi Jalan

Gambar 2.4 menunjukkan hubungan antara koefisien gesekan (μ_r) beberapa kondisi jalan dengan besar nilai *slip*(λ) pada kendaraan. Nilai *slip*(λ) yang ingin dipertahankan adalah diantara 0,18-0,22. Berikut merupakan persamaan matematis *slip*(λ) pada kendaraan:

$$\lambda = \frac{v_x - \omega R}{v_x} \tag{2.4}$$

Di mana v_x merupakan kecepatan longitudinal kendraaan (m/s), ω merupakan kecepatan sudut roda (rad/s), dan R merupakan jari-jari roda kendaraan (m).

2.4 Rem Elektromagnetik [4]

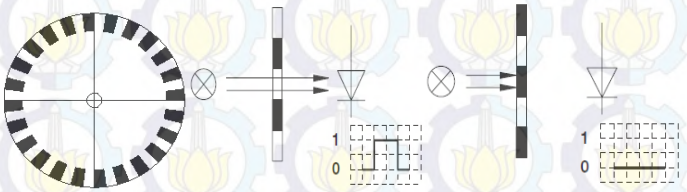
Sistem pengereman ini menggunakan gaya elektromagnetik untuk memperlambat suatu gerakan, yang umumnya adalah gerakan poros. Sebuah piringan dengan bahan logam non-ferromagnetik terpasang disebuah poros berputar. Piringan tersebut diapit oleh sisi stator berupa sistem lilitan elektromagnetik yang dapat membangkitkan medan magnet dari aliran listrik. Arus listrik menimbulkan medan magnet pada lilitan. Kemudian logam piringan yang memotong medan magnet tersebut akan menimbulkan arus *eddy* pada piringan itu sendiri. Arus *eddy* ini akan menimbulkan medan magnet yang arahnya berlawanan dengan medan magnet sebelumnya, sehingga menghambat gerakan putar dari poros tersebut. Jenis yang digunakan pada tugas akhir ini adalah jenis *micro electromagnetic brake*.

Gambar 2.6 *Micro Electromagnetic Brake*

2.5 *Rotary Encoder* [5]

Pada Tugas Akhir ini, peneliti menggunakan dua buah *rotary encoder* untuk mengukur kecepatan putar dan kecepatan *longitudinal* pada simulator ABS. *Rotary encoder* adalah suatu komponen elektro mekanis yang memiliki fungsi untuk memonitoring posisi anguler pada suatu poros yang berputar. Dari perputaran benda tersebut data yang termonitoring akan diubah ke dalam bentuk data digital oleh *rotary encoder* berupa lebar pulsa, kemudian akan dihubungkan ke kontroler (Mikrokontroler/PLC). Berdasarkan data yang di dapat berupa posisi

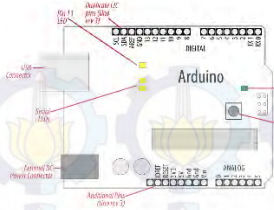
anguler (sudut) kemudian dapat diolah oleh kontroler sehingga mendapatkan data berupa kecepatan, arah, dan posisi dari perputaran porosnya. *Rotary encoder* menggunakan sensor optik untuk menghasilkan pulsa listrik sehingga sudut, posisi, kecepatan, dan percepatan suatu benda dapat dihitung. *Rotary encoder* menggunakan piringan kaca atau plastik yang memiliki lubang pada sekeliling lingkaran. Pada *rotary encoder, Light Emitting Dioda* (LED) ditempatkan pada salah satu sisi piringan dan *phototransistor* diletakan pada sisi yang berseberangan sehingga dapat mendeteksi cahaya dari LED. Pada saat piringin berputar, *phototransistor* mendeteksi ada atau tidaknya cahaya dari LED. Ketika sumber cahaya dapat melewati piringan dan dideteksi oleh *phototransistor, phototransistor* akan mengeluarkan pulsa listrik 5 Volt. Sebaliknya apabila piringan menutup jalur cahaya dari sumber cahaya menuju *phototransistor*, maka *phototransistor* akan mengeluarkan pulsa listrik 0,5 Volt. Cara kerja *rotary encoder* dapat dilihat pada Gambar 2.6 berikut:



Gambar 2.7 Cara Kerja *Rotary Encoder*

2.6 Arduino Uno [6]

Arduino adalah *platform* pembuatan *prototype* elektronik yang bersifat *open-source hardware* yang berdasarkan pada perangkat keras dan perangkat lunak yang fleksibel dan mudah digunakan. *Platform* arduino terdiri dari arduino *board*, *shield*, bahasa pemrograman arduino, dan arduino *development environment.* Arduino *board* biasanya memiliki sebuah *chip* dasar mikrokontroler Atmel AVR ATmega8 berikut turunannya.



Gambar 2.8 Arduin Uno

Arduino memiliki *Universal Serial Bus* (USB) *serial port* yang menggantikan 9 pin RS-232 serial konektor melalui USB *interface chips*. Arduino menyediakan 2 jenis *input power supply*, yaitu melalui USB *serial port* dan melalui *jack* DC *voltage* dengan tegangan *input* yang disarankan 7 – 12 Volt, dan batas tegangan *input*nya adalah 6 – 20 Volt. Arduino menyediakan tegangan 5 Volt melalui USB *serial port* kepada IC ATmega328, sehingga mikrokontroler yang dipakai dapat berjalan pada *clock-rate* maksimum (20 MHz). Arduino memiliki 28 buah pin, dimana 20 pin digunakan untuk pin *input*/*output* (14 pin untuk *digital input/output* dan *analog input* berupa PWM, serta 6 pin untuk analog *input*), 7 pin suplai tegangan (5 Volt, 3,3 Volt, dan *Ground*), dan 1 buah pin *reset*. Arus DC yang dapat dikeluarkan oleh Arduino adalah 40 mA pada tegangan 5 Volt, dan 50 mA pada tegangan 3,3 Volt. Arduino memiliki memori untuk data sebesar 2 *Kbytes* dan *Electrically Erasable Programmable Read-Only Memory* (EEPROM) sebesar 1 *KBytes*. Arduino juga memiliki *In Circuit Serial Programming (*ICSP) *header*, dan sebuah tombol *reset*.

2.7 Identifikasi Sistem [7]

Sebelum kita dapat melakukan pengaturan proses suatu *plant*, terlebih dahulu kita harus mengetahui dinamika dari *plant* tersebut. Dinamika dari suatu *plant* dapat direpresentasikan dalam suatu persamaan matematis yang sering disebut dengan model matematika sistem. Salah satu bentuk model matematika *plant* yang paling sering digunakan adalah *transfer function*. Model matematika dari sebuah sistem dapat diperoleh melalui dua cara yaitu permodelan fisik dan identifikasi sistem. Permodelan fisik merupakan pendekatan matematis hukum – hukum matematika untuk menjelaskan dinamika dalam *plant*. Sedangkan identifikasi sistem adalah usaha untuk mendapatkan model parameter *plant* berdasarkan data *input* dan *output plant*. Proses identifikasi sistem

dapat dilakukan secara *online* maupun secara *offline*. Pada beberapa kasus yang sangat kompleks, sangat sulit untuk menentukan model berdasarkan identifikasi sistem.

Identifikasi sistem pada suatu plant dapat dilakukan dengan menggunakan metode identifikasi statis maupun dinamis. Identifikasi statis adalah proses identifikasi *plant* dimana sinyal *input* yang diberikan konstan terhadap waktu, sedangkan identifikasi dinamis menggunakan sinyal *input* dengan nilai yang beragam terhadap waktu. Biasanya sinyal uji yang digunakan pada identifiksai statis adalah sinyal *step*, sedangkan pada identifikasi dinamis adalah sinyal *random*. Pada Tugas Akhir kali ini, digunakan identifikasi dinamis dengan pemodelan sistem deterministik

Untuk melakukan proses identifikasi sistem tersebut diperlukan langkah-langkah sebagai berikut:

- Pengambilan data *input* dan *output*
- Menentukan struktur model
- Estimasi parameter
- Validasi model

2.7.1 Pengambilan Data *Input* dan *Output*

Langkah awal dalam melakukan identifikasi sistem adalah pengambilan data *input-output*. Pengujian ini tentu memerlukan sinyal uji tertentu yang akan diberikan kepada sistem fisik yang akan diidentifikasi. Agar diperoleh model yang tepat maka dalam pemilihan sinyal uji tidak boleh sembarangan. Syarat pemilihannya adalah sinyal uji harus memiliki cakupan frekuensi yang lebar:

2.7.2 Menentukan Struktur Model

Secara umum struktur dalam identifikasi adalah sebagai berikut:

$$A(q)y(k) = \frac{B(q)}{F(q)}u(k-nk) + \frac{C(q)}{D(q)}\eta(k) \quad (2.5)$$

Di mana :

$$A(q) = a_1q^{-1} + a_2q^{-2} + \cdots + a_{na}q^{-na} \quad (2.6)$$

$$B(q) = b_1q^{-1} + b_2q^{-2} + \cdots + b_{nb}q^{-nb}) \quad (2.7)$$

$$C(q) = c_1q^{-1} + c_2q^{-2} + \cdots + c_{nc}q^{-nc} \quad (2.8)$$

$$D(q) = d_1q^{-1} + d_2q^{-2} + \cdots + d_{nd}q^{-nd} \quad (2.9)$$

$$F(q) = f_1q^{-1} + f_2q^{-2} + \cdots + f_{nf}q^{-nf} \quad (2.10)$$

$$y(k) = output \quad (2.11)$$

$$u(k) = input \quad (2.12)$$

$$\eta(k) = noise \quad (2.13)$$

Berdasarkan Persamaan 2.5 terdapat dua bentuk pemodelan sistem diskrit, yaitu bentuk deterministik dan bentuk stokastik. Bentuk deterministik adalah pemodelan sistem diskrit yang tidak mengikutsertakan *noise* dalam perhitungan estimasi parameter model, sedangkan bentuk stokastik adalah pemodelan sistem diskrit yang mengikutsertakan *noise* dalam perhitungan estimasi parameter. Berdasarkan Persamaan 2.5 juga dapat dibentuk pemodelan sistem diskrit yang bergantung pada adanya polinomial A, B, C, D, dan F yang dapat dilihat pada Tabel 2.1 berikut:

Tabel 2.1 Macam-Macam Struktur Model

Struktur	Bentuk Pemodelan
A	*Auto-Regressive* (AR)
B	*Moving Average* (MA)
AB	*Auto-Regressive Moving Average* (ARMA)
AC	*Auto-Regressive with eXogenous input* (ARX)
BC	*Moving Average with eXogenous input* (MAX)
ABC	*Auto-Regressive Moving Average with eXogenous input* (ARMAX)
ABD	*ARARX*
BF	*Output-Error (OE)*
BFCD	*Box-Jenkins (BJ)*

2.7.3 Metode Estimasi Parameter

Untuk mengestimasi parameter model, sebuah kriteria harus dinyatakan terlebih dahulu. Kriteria yang digunakan pada tugas akhir ini adalah *least square.* Metode *least square* merupakan pengembangan dari metode *gradient* dengan kriteria yang diminimumkan melibatkan semua pengukuran yang dapat dilihat pada Persamaan 2.14 berikut:

$$min\, J = min\left\{\frac{1}{2}\sum_{i=0}^{k}\left[y(i) - \varphi^T(i-1)\hat{\theta}(i)\right]^2\right\} \quad (2.14)$$

Metode *least square* memiliki dua algoritma dalam mendapatkan nilai estimasi parameter $\hat{\theta}(k)$ yaitu algoritma *Standart Least Square* (SLS) dan algoritma *Extended Least Square* (ELS). Algoritma SLS terdiri

dari proses inialisasi dan proses iterasi. Proses inialisasi diawali dengan penentuan bentuk model dan orde sistem. Selanjutnya adalah penetuan kondisi awal $\varphi(k-1)$, $\theta(k-1)$, dan *gain* estimasi. Lalu proses iterasi dilakukan dengan mengukur *input output* dari *plant*, dan dilanjutkan dengan rekonstruksi $\varphi^T(k-1)$. Selanjutnya menghitung nilai estimasi parameter $\hat{\theta}(k)$ dan merevisi nilai *gain* estimasi. Jika proses iterasi sudah selesai maka nilai estimasi parameter terakhir adalah parameter yang dibutuhkan dalam pemodelan sistem. Pada metode ELS memiliki algoritma yang sama dengan metode SLS yang berbeda hanya pada penghitungan revisi nilai *gain* estimasi

2.7.4 Validasi Model

Validasi model digunakan untuk membedakan model yang akurat terhadap model yang kurang akurat. Keakurasian model diuji dengan cara membandingkan respon model dengan respon sistem yang sebenarnya terhadap sinyal *input* tertentu seperti; sinyal *step,* kotak, dan PRBS (*Pseudo Random Binary Sequences).* Salah satu metode validasi model adalah metode *Root Mean Square Error* (RMSE). RMSE mengukur akurasi pada nilai deret waktu secara statistik seperti halnya regresi. RMSE dapat merepresentasikan ukuran dari *error* rata-rata karena RMSE membandingkan hasil data pengukuran dan data pemodelan pada skala yang sama antara kedua data tersebut. Metode RMSE dapat ditulis dalam persamaan matematis sebagai berikut:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.15}$$

dimana *n* adalah jumlah data, y_i adalah hasil identifikasi dinamis *plant,* dan $\hat{y}_i$ adalah hasil pemodelan. Jika nilai hasil pengukuran RMSE semakin kecil maka hasil pemodelan sistem dapat dikatakan sudah baik.

2.8 Kontroler [8]

Dalam sebuah sistem kontrol, kontroler mempunyai peran penting terhadap perilaku sistem. Pada prinsipnya hal ini disebabkan oleh tidak dapat diubahnya komponen penyusun sistem tersebut. Artinya karakteristik *plant* harus diterima sebagai mana adanya, sehingga perilaku sistem hanya dapat diperoleh melalui penambahan suatu subsistem.

Salah satu kontoler yang sering digunakan di industri adalah kontoler PID. kontoler PID adalah kontroler yang memanfaatkan sinyal

error sistem yang akan diolah oleh tiga elemen kontroler, yakni proporsional, integral, dan derivatif. Tiga elemen dalam kontroler PID tersebut dapat dikombinasikan menjadi beberapa tipe kontroler sesuai dengan kebutuhan *plant*. Masing-masing elemen memiliki pengaruh yang berbeda pada sistem. Berikut merupakan Pengaruh K_p, K_i, K_d pada *closed loop* respon:

Tabel 2.2 Pengaruh K_p, K_i, K_d pada Respon Sistem

Respon *Close-Loop*	*Rise Time*	*Overshoot*	*Setting Time*	*SS Errror*
K_p	Turun	Naik	Perubahan Kecil	Turun
K_i	Turun	Naik	Naik	Hilang
K_d	Perubahan Kecil	Turun	Turun	Perubahan Kecil

Dari Tabel di atas dapat diketahui bahwa pengendali proporsional akan mengurangi waktu naik, meningkatkan persentase lewatan maksimum dan mengurangi keadaan tunak. Sedangkan pengendali proporsional derivatif mereduksi lewatan maksimum dan waktu turun. Selain itu, pengendali proporsional integral menurun pada waktu naik, meningkatkan lewatan maksimum dan waktu turun dan akan menghilangkan kesalahan keadaan. Salah satu permasalahan terbesar dalam desain kontroler PID yaitu masalah *tuning* untuk menentukan nilai Ki, Kp, dan Kd yang tepat. Metode –metode tuning dilakukan berdasarkan model matemetika *plant* sistem. Jika model tidak diketahui, maka dilakukan eksperimen terhadap sistem. Dapat juga memakai sistem *trial* dan *error*. Kontroler PID ini merupakan jenis kontroler yang paling populer digunakan yang banyak diterapkan di dunia industri.

2.8.1 Kontroler Tipe Proporsional

Kontroler tipe-P merupakan kontroler yang hanya terdiri dari elemen proporsional saja. Hubungan antara *input* sinyal *error* $e(t)$ dengan sinyal kontrol $u(t)$ pada kontroler proporsional adalah sebagai berikut:

$$u(t) = K_p e(t) \tag{2.16}$$

atau dalam bentuk *transfer function* dapat dituliskan sebagai berikut:

$$\frac{U(s)}{E(s)} = K_p \tag{2.17}$$

dimana K_p adalah penguatan proporsional. K_p berfungsi untuk memepercepat kecepatan respon sistem.Semakin besar nilai K_p, maka semakin cepat respon sistem. Namun penggunaan nilai K_p yang terlalu besar dapat menyebabkan *overshoot* pada sistem, dan jika nilai K_p terlalu kecil, maka settling time sistem akan lambat.

2.8.2 Kontroler Tipe Proporsional Integral

Kontroler tipe-PI merupakan kontroler yang terdiri dari elemen proporsional dan integral. Hubungan antara *input* sinyal *error* $e(t)$ dengan sinyal kontrol $u(t)$ pada kontroler tipe-PI adalah sebagai berikut:

$$u(t) = K_p\left(e(t) + \frac{1}{\tau_i}\int_0^t e(t)dt\right) \tag{2.18}$$

atau dalam bentuk *transfer function* dapat dituliskan sebagai berikut:

$$\frac{U(s)}{E(s)} = K_p\left\{1 + \frac{1}{\tau_i s}\right\} = K_p + \frac{K_i}{s} \tag{2.19}$$

dimana K_i adalah penguatan integral dan τ_i adalah waktu integral. K_i berfungsi untuk menghilangkan *steady-state error*. Namun apabila nilai $K_i$terlalu besar, maka dapat menimbulkan osilasi pada sinyal *output* sistem.

2.8.3 Kontroler Tipe Proporsional Derivatif

Kontroler tipe PD merupakan kontroler yang hanya terdiri dari elemen proporsional dan derivatif. Hubungan antara *input* sinyal *error* $e(t)$ dengan sinyal kontrol $u(t)$ pada kontroler tipe-PD adalah sebagai berikut:

$$u(t) = K_p\left(e(t) + \tau_d\frac{de(t)}{dt}\right) \tag{2.20}$$

Atau dalam bentuk *transfer function* dapat dituliskan sebagai berikut:

$$\frac{U(s)}{E(s)} = K_p\{1 + \tau_d s\} = K_p + K_d s \quad (2.21)$$

Di mana K_d adalah penguatan deferensial dan τ_d adalah waktu derivative. K_d berfungsi untuk mengurangi *overshoot* dan meningkatkan kestabilan sistem. Ketika nilai K_d terlalu besar, maka *settling time output* sistem akan sangat lama.

2.8.4 Kontroler Tipe PID

Kontroler tipe PID merupakan kontroler yang memiliki elemen proporsional, integral, dan derivatif. Hubungan antara *input* sinyal *error* $e(t)$ dengan sinyal kontrol $u(t)$ pada kontroler proporsional adalah sebagai berikut:

$$u(t) = K_p\left(e(t) + \frac{1}{\tau_i}\int_0^t e(t)dt + \tau_d\frac{de(t)}{dt}\right) \quad (2.22)$$

Atau dalam bentuk *transfer function* dapat dituliskan sebagai berikut:

$$\frac{U(s)}{E(s)} = K_p\left\{1 + \frac{1}{\tau_i s} + \tau_d s\right\} = K_p + \frac{K_i}{s} + K_d s \quad (2.23)$$

K_p adalah penguatan proporsional. K_p berfungsi untuk memepercepat kecepatan respon sistem.Semakin besar nilai K_p, maka semakin cepat respon sistem. K_i adalah penguatan integral dan τ_i adalah waktu integral. K_i berfungsi untuk menghilangkan *steady-state error*. K_d adalah penguatan deferensial dan τ_d adalah waktu derivative. K_d berfungsi untuk mengurangi *overshoot* dan meningkatkan kestabilan sistem.

2.9 Metode Penentuan Parameter PID [9]

Penentuan parameter PID kontroler berdasarkan pada metode perhitungan analitik dengan pendekatan respon orde I. Pada perancangan kontroler PID secara analitik memerlukan beberapa tahapan pekerjaan,yaitu:

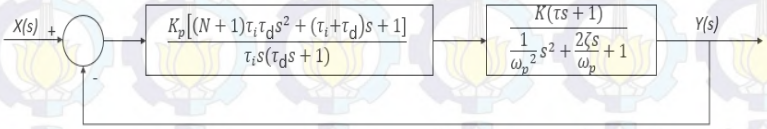
a. Menentukan model matematik *plant*, model matematik *plant* dapat diturunkan melalui hubungan fisik antar komponen atau dengan menggunakan metode identifikasi.

b. Menentukan spesifikasi performansi, karena perancangan ini tergolong dengan pendekatan respon waktu dan hanya untuk sistem orde I dan orde II saja, maka ukuran kualitas respon yang digunakan adalah ukuran kualitas respon waktu. Biasanya digunakan *settling time dan %error steady state* untuk pendekatan respon orde I, atau *settling time, %overshoot* dan *%error steady state* untuk pendekatan respon orde II.

c. Merancang kontroler PID, adalah tahapan akhir dari perancangan yang meliputi pemilihan tipe kontroler dan menghitung nilai parameter kontroler.

2.9.1 Perancangan PID Modifikasi untuk *Plant* Orde II dengan *Delay*

Untuk merancang kontroler PID dengan respon *output* menyerupai orde I, maka dirancang kontroler PID dengan metode analitik dengan cara sebagai berikut:



Gambar 2.9 Blok Diagram Modifikasi Kontroler Tipe PID untuk *Plant* Orde II dengan *Delay*

Perancangan PID ini bertujuan untuk mendapatkan parameter PID sehingga respon *output* sistem mendekati orde I. *Closed loop transfer function* dari sistem dapat dinyatakan dalam Persamaan 2.24:

$$\frac{Y(s)}{X(s)} = K_p K[(N+1)\tau_i\tau_d s^2+(\tau_i+\tau_d)s+1](\tau s+1)/1$$
$$\{\tau_i s(\tau_d s+1)\left(\frac{1}{\omega_p{}^2}s^2+\frac{2\zeta s}{\omega_p}+1\right)+K_p K(\tau s+1) \quad (2.24)$$
$$[(N+1)\tau_i\tau_d s^2+(\tau_i+\tau_d)s+1]\}$$

Di mana:
Y(s) = *output*
X(s) = *input*
K = *gain overall*
Kp = konstanta penguatan proporsional

N = konstanta *delay*
τ_i = waktu integral
τ_d = waktu derivatif
ω_p = frekuensi natural
ζ = konstanta redaman

Dari Persamaan 2.24, tampak bahwa orde dari sistem hasil desain tergantung pada pemilihan τ_i, τ_d dan N. Jika dipilih $\tau_d = \tau$ dan $\tau_i + \tau_d = 2\zeta/\omega_p$serta $(N+1)\tau_i\tau_d = {\omega_p}^{-2}$, maka sistem hasil desain adalah orde I. Sebaliknya jika nilai τ_i, τ_d dan N tidak dipilih seperti nilai diatas, maka sistem hasil desain dapat berupa orde II, orde III atau orde IV. Agar hasil desain adalah orde I, maka dipilih $\tau_d = \tau$ dan $\tau_i + \tau_d = 2\zeta/\omega_p$serta $(N+1)\tau_i\tau_d = {\omega_p}^{-2}$ sehingga CLTF sistem hasil desain menjadi:

$$\frac{Y(s)}{X(s)} = \frac{K_p K}{\tau_i S + K_p K} \tag{2.25}$$

Atau

$$\frac{Y(s)}{X(s)} = \frac{1}{\tau^* S + 1} \tag{2.26}$$

Dengan

$$\tau^* = \frac{\tau_i}{K_p K} \tag{2.27}$$

Berdasarkan hubungan formulasi di atas, parameter kontroler dapat dituliskan sebagai:

$$\tau_d = \tau \tag{2.28}$$

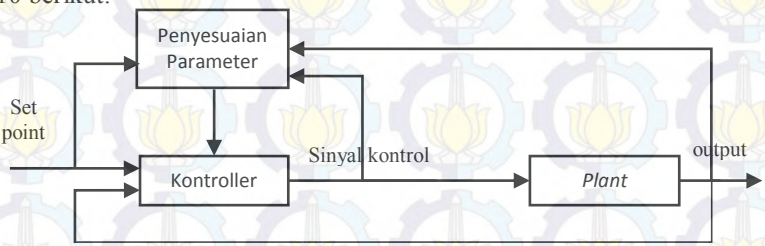
$$\tau_i = \frac{2\zeta}{\omega_p} - \tau \tag{2.29}$$

$$N = \frac{1}{\tau\omega_p(2\zeta - \tau\omega_p)} - 1 \tag{2.30}$$

$$\tau_i = \frac{2\zeta}{\omega_p} - \tau \tag{2.31}$$

2.10 Kontroler Adaptif [10]

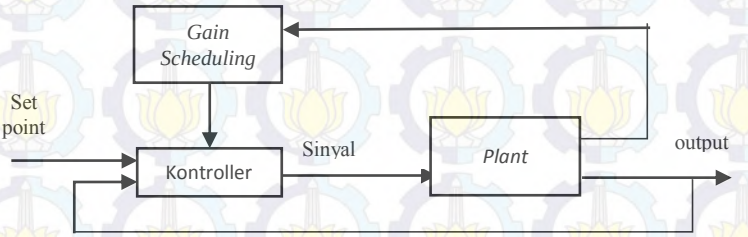
Kontrol adaptif adalah suatu kontrol yang dapat memodifikasi tingkah laku respon untuk mengubah proses dan karakteristik gangguan secara dinamis. Diagram blok kontroler adaptif dapat dilihat pada Gambar 2.10 berikut.



Gambar 2.10 Diagram Kontrol Adaptif

2.10.1 *Gain Scheduling*

Gain Scheduling adalah nonlinier *feedback* dimana parameter kontroler linier berubah fungsi setiap kondisi operasi. *Gain Scheduling* berdasarkan pada perngukuran kondisi operasi suatu proses yang sering digunakan untuk menkompensasi suatu variasi proses nonlinier. *Gain Scheduling* teknik yang sangat membantu unutk mengurangi efek dari variasi parameter.

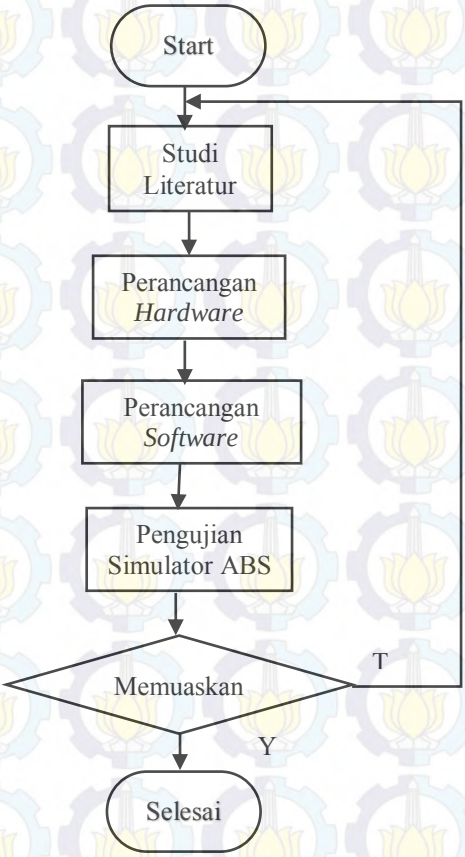


Gambar 2.11 Diagram Blok *Gain Scheduling*

Ketika variabel *scheduling* telah ditentukan, parameter kontroler akan dikalkulasikan di kondisi operasi dengan menggunakan metode desain yang cocok. Kontroler perlu dikalibrasi untuk setiap kondisi operasi. Stabilitas dan performansi dari sistem dapat dievaluasi dari simulasi.

BAB 3
PERANCANGAN SISTEM

Pada bab ini akan dibahas mengenai perancangan kontroler PID adaptif untuk mempertahankan rasio *slip* yaitu sekitar 20% dan Gambar 3.1 menerangkan tentang yang harus dilakukan untuk menyelesaikan tugas akhir ini.



Gambar 3.1 Diagram Alir Implementasi

Dari Gambar 3.1 maka perancangan dan implementasi kontrol PID adaptif untuk pengaturan *slip* dan rasio percepatan terbagi dalam beberapa tahap.

Tahap pertama yaitu mempelajari teori secara keseluruhan sistem, kebutuhan sistem, dan hal lainnya dari berbagai literatur yang sudah ada. Setelah itu dilanjutkan dengan merancang *hardware* yang dibutuhkan.

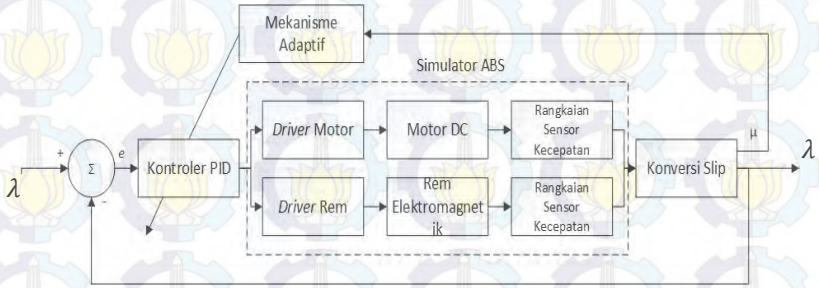
3.1 Gambaran Umum Sistem

Simulator ABS (*Anti-Lock Braking System*) merupakan peralatan untuk melakukan pengujian dan pengoptimalan kinerja metode kontrol *slip*. Simulator ABS ini memodelkan kecepatan mobil yang hanya memperhatikan gerakan *longitudinal* dari mobil dan gerakan *angular* dari ban selama proses pengereman. Simulator ABS ini menggunakan dua buah roda yang diletakkan secara vertikal dan saling bersinggungan. Roda bawah dianggap sebagai gerakan jalan (kecepatan *longitudinal*) dan roda atas dianggap sebagai gerakan pada mobil. Untuk mendapatkan *slip* rasio dari simulator ABS, dilakukan beberapa pengujian kecepatan dan pengereman.

Pada simulator ini, roda atas dianggap sebagai roda bebas sehingga tidak diberi pemberat. Aktuator yang digunakan pada simulator ini berupa rem mikro elektromagnetik yang dipasang pada roda atas dan diberi sinyal *Pulse Width Modulation* (PWM) agar dapat bekerja. Sebelum melakukan proses pengereman, maka roda atas harus memiliki kecepatan putar tertentu. Agar roda atas memiliki kecepatan putar tertentu, maka roda bawah yang bersinggungan dengan roda atas, dihubungkan dengan motor *Direct Current* (DC) yang diberi sinyal PWM tertentu agar dapat berputar.

Proses pengereman dapat dilakukan dengan cara menghilangkan suplai sinyal PWM ke motor dan mulai memberikan sinyal PWM ke rem elektromagnetik yang terpasang pada roda atas. Torsi rem yang ditambahkan pada roda bagian atas menyebabkan perlambatan pada roda atas dan roda bawah. Akan tetapi, karena terdapat perbedaan kelembaman, maka terjadi perbedaan perlambatan atau penurunan kecepatan pada kedua roda. Perbedaan kecepatan roda atas dan roda bawah ini menghasilkan *slip* yang dapat dihitung. Selain perbedaan kelembaman anatar kedua roda yang mempengaruhi besarnya *slip* yang terjadi, ada beberapa parameter juga yang mempengaruhi besarnya *slip* pada simulator ABS, seperti koefisien permukaan roda, dan besarnya torsi

pengereman yang diberikan pada roda atas. Diagram blok pengaturan *slip* dapat dilihat pada Gambar 3.2.



Gambar 3.2 Blok Diagram Sistem ABS

Dapat dilihat pada Gambar 3.2 sistem ABS terdiri dari *input* berupa *slip* yang diharapkan, kontroler berupa kontroler PID, *interface* arduino uno, simulator ABS, dan *output* berupa *slip* dan pengoperaian kondisi(μ). Hasil kalkulasi pada arduino uno adalah data yang didapatkan dari sensor kecepatan roda atas dan roda bawah. *plant* simulator ABS berupa roda atas yang dihubungkan dengan rem elektromagnetik, dan roda bawah yang dihubungkan dengan motor DC. Aktuator pada Simulator ABS adalah *driver* rem elektromagnetik yang mengatur besarnya persentase rem. Alur kerja diagram blok sistem ABS yakni Arduino uno akan mengirimkan nilai *slip* melalui komunikasi serial ke komputer, kemudian komputer akan mengolah sinyal *error* menggunakan kontroler PID pada program matlab. Mekanisme adaptif dalam sistem ABS akan memilih kontroler yang akan digunakan pada *plant* sehingga respon *output* dari *plant* sesuai dengan respon yang diinginkan. Mekanisme adaptif didasarkan pada kondisi operasi μ yang dihasilkan dari percepatan pada roda bawah.

3.2 Perancangan *Hardware*

Pada tahap ini dilakukan 2 jenis perancangan, yaitu perancangan mekanik dan perancangan elektronik. Perancangan mekanik merupakan perancangan dengan menempatkan komponen-komponen yang digunakan dalam simulator ABS, sehingga dapat digunakan untuk melakukan pengujian kontrol *slip*. Sedangkan perancangan elektronik merupakan perancangan pembuatan *driver* dan rangkaian sensor.

3.2.1 Perancangan Mekanik

Perancangan mekanik pada simulator ABS dibagi menjadi 2 bagian, yaitu perancangan bagian atas dan perancangan bagian bawah. Perancangan bagian atas meliputi komponen-komponen yang digunakan untuk menghasilkan kecepatan pada mobil, dan bagian bawah meliputi komponen-komponen yang digunakan untuk menghasilkan kecepatan *longitudinal*.

3.2.1.1 Perancangan Bagian Atas

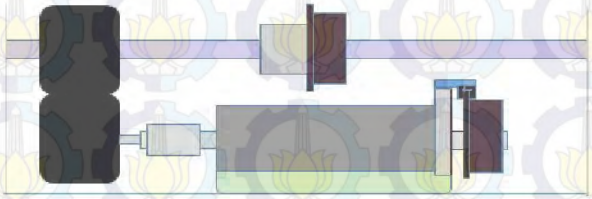
Perancangan mekanik pada bagian atas meliputi penempatan roda, rem mikro elektromagnetik, dan sensor *rotary encoder* pada satu *shaft*. Roda yang dipasang merepresentasikan roda mobil yang berputar. Roda ini dianggap sebagai roda bebas sehingga tidak dihubungkan secara langsung dengan motor DC, melainkan dihubungkan dengan roda bagian bawah secara bersinggungan agar roda atas dapat berputar dan dilakukan proses pengereman dengan metode kontrol *slip* tertentu.

Pada *shaft* dipasang sebuah sensor *rotary encoder* dan rem mikro elektromagnetik. Sensor *rotary encoder* terdiri dari piringan *encoder* dan rangkaian sensor *optocoupler*. Piringan *encoder* terbuat dari bahan plastik dan memiliki 55 lubang. Rem mikro elektromagnetik dipasangkan pada bagian sisi dalam piringan *encoder*, sehingga pada saat rem bekerja timbul arus eddy dan gaya yang berlawanan pada piringan *encoder*. Gaya yang ditimbulkan menyebabkan perlambatan pada roda bebas. Perlambatan tersebut menyebabkan *slip* antara roda bebas dan roda bagian bawah.

3.2.1.2 Perancangan Bagian Bawah

Perancangan mekanik pada bagian bawah simulator ABS meliputi penempatan roda, penempatan sensor *rotary encoder* pada *shatf* motor DC. Roda yang berputar pada bagian bawah merepresentasikan kecepatan *longitudinal* mobil, maa dari itu roda ini memiliki massa lebih besar daripada roda bebas.

Pada bagian ujung *shaft* motor DC dipasang sebuah sensor *rotary encoder*. Sensor *rotary encoder* terdiri dari piringan *encoder* dan rangkaian sensor *optocoupler*. Piringan *encoder* terbuat dari bahan plastik dan memiliki 55 lubang. Rangkaian sensor *optocoupler* dipasangkan pada rangka motor DC, dan piringan *encoder* dipasangkan pada bagian tengah sensor *optocoupler*.



Gambar 3.3 Sketsa Tampak Depan Simulator ABS

3.2.2 Perancangan Elektronik

Perancangan elektronik pada simulator ABS yang digunakan untuk Tugas Akhir ini dibagi menjadi empat bagian yaitu perancangan *driver* motor DC, *driver* rem mikro elektromagnetik, rangkaian sensor *optocoupler*, arduino UNO.

3.2.2.1 Driver Motor DC

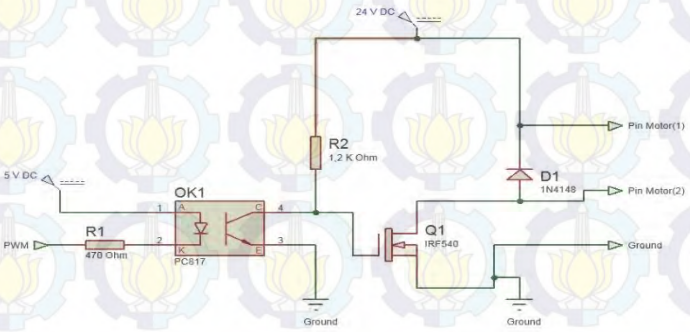
Rangkaian *driver* motor DC digunakan untuk menjalankan motor DC yang menjadi penggerak utama dalam simulator ABS. motor DC yang digunakan adalah motor DC buatan Nisca *Corporation* dengan spesifikasi yang dapat dilihat pada Tabel 3.1. *Driver* motor DC ini terdiri dari sebuah *optocoupler* PC817, *N-Channel Power MOSFET* IRF240, dioda 1N414800, dan resistor 1,2 KOhm beserta 470 Ohm. Tegangan *input* agar *driver* ini dapat menyala adalah 5 Volt. Sedangkan tegangan *input driver* ini adalah PWM Arduino yang akan diubah menjadi PWM 24 Volt sebagai tegangan *output*nya. Pada rangkaian driver ini, sinyal *input* PWM Arduino dihubungkan dengan Arduino UNO di pin *digital* 10.

Cara kerjanya adalah mula-mula arduino memberikan sinyal *Pulse Width Modulation* (PWM) ke rangakaian motor DC. Ketika arduiono memberikan pulsa *High* maka *Light Emitting Dioda* (LED) *optocupler* tidak akan menyala sehingga *transistor optocoupler* menjadi *open* dan tegangan Vcc motor mengalir ke MOSFET. Akibat MOSFET mendapatkan tegangan Vcc motor maka motor DC dapat menyala. Pada saat Arduino memberikan pulsa *low* maka *Light Emitting Dioda* (LED) *optocoupler* akan menyala sehingga *transistor optocoupler* menjadi *short*

dan tegangan Vcc motor mengalir ke *optocoupler*. Akibat MOSFET tidak mendapatkan tegangan Vcc motor maka motor DC tidak dapat menyala. Dioda pada rangkaian *driver* motor digunakan sebagai penyearah tegangan bolak balik akibat *optocoupler* sehingga MOSFET tidak terlalu bekerja keras atau cepat panas.

Tabel 3.1 Spesifikasi Motor DC yang Digunakan [11]

Tipe motor DC	*Brush*
Tegangan operasi motor DC	0-24Volt
Bearing motor DC	*Ball bearing*
Berat motor DC	700 gram
Magnet motor DC	2 *pole* magnet permanen
Kecepatan motor DC	2500 rpm
Arus motor DC	1,123 A
Daya *output* motor DC	19,15 W
Efisiensi motor DC	71,1%



Gambar 3.4 Rangkaian *Driver* Motor DC

Gambar 3.4 merupakan skematik rangkaian *driver* motor DC yang dibuat pada *software* proteus. Rangkaian menggunakan berberapa komponen yakni, konektor ptr 500, ic *optocopler* pc 817, resistor 1,2 KOhm, resistor 470 Ohm dioda *clamp*, dan transistor *FET IRF* 540.

3.2.2.2 *Driver* Rem Mikro Elektromagnetik

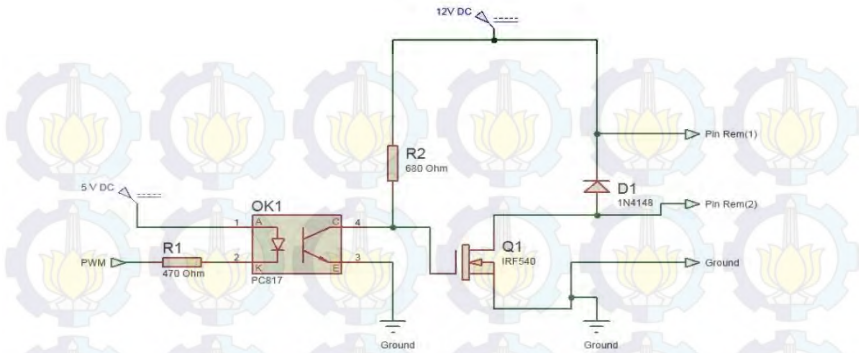
Rangkaian *driver* rem mikro elektromagnetik digunakan untuk menjalankan rem yang menjadi aktuator dalam metode kontrol *slip* di

simulator ABS pada Tugas Akhir ini. Adapun rem yang digunakan adalah rem mikro elektromagnetik buatan Shinko Electro dengan spesifikasi yang dapat dilihat pada Tabel 3.2. *Driver* rem mikro elektormagnetik ini terdiri dari sebuah *optocoupler* PC817, *N-Channel Power MOSFET* IRF240, dioda 1N414800, dan resistor 680Ohm beserta 470 Ohm. Tegangan *input* agar *driver* ini dapat menyala adalah 5 Volt. Sedangkan tegangan *input driver* ini adalah PWM Arduino yang akan diubah menjadi PWM 12 Volt sebagai tegangan *output*nya. Pada rangkaiana *driver* ini, sinyal *input* PWM Arduino dihubungkan dengan Arduino UNO di pin *digital* 9.

Cara keja dari rangkaian *driver* rem mikro elektromagnetik ini adalah Arduino memberikan sinyal PWM ke rangkaian rem mikro elektromagnetik. Ketika Arduino memberikan pulsa *high* maka *Light Emitting Dioda* (LED) *optocoupler* tidak akan menyala sehingga *transistor optocoupler* menjadi *open* dan tegangan Vcc rem mengalir ke MOSFET. Akibat MOSFET mendapatkan tegangan Vcc rem maka rem mikro elektromagnetik dapat menyala. Pada saat Arduino memberikan pulsa *low* maka *Light Emitting Dioda* (LED) *optocoupler* akan menyala sehingga *transistor optocoupler* menjadi *short* dan tegangan Vcc rem mengalir ke *optocoupler*. Akibat MOSFET tidak mendapatkan tegangan Vcc rem maka rem mikro elektromagnetik tidak dapat menyala. Dioda pada rangkaian *driver* rem mikro elektromagnetik digunakan sebagai penyearah tegangan bolak balik yang diakibatkan *optocoupler* sehingga MOSFET tidak terlalu bekerja keras atau cepat panas.

Tabel 3.2 Spesifikasi Rem Mikro Elektromagnetik **[12]**

Tegangan pengoprasian rem	0-12 Volt
Daya pengoperasian rem	3,3 Watt
Produksi	Shinko Electro
Jenis	Kering dengan satu piringan (*dry-type single-plate*)
Ukuran diameter rem	Luar: 2,7 cm Dalam: 1,1cm
Tinggi rem	1,5 cm



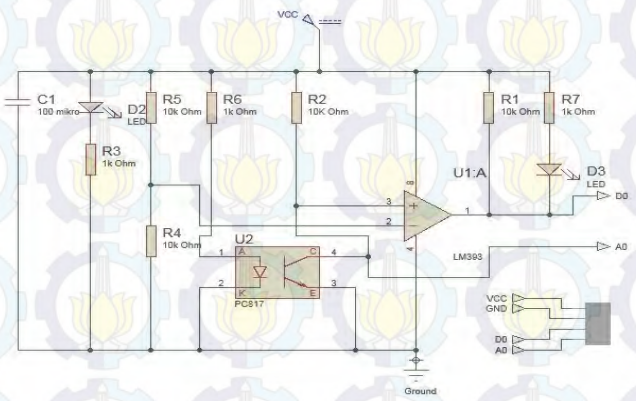
Gambar 3.5 Rangkaian *Driver* Rem Mikro Elektromagnetik

Gambar *skematik* rangkaian *driver* motor DC dapat dilihat pada Gambar 3.5. Skematik rangkaian *driver* rem dibuat pada *software* proteus. Rangkaian menggunakan berberapa komponen yakni, konektor ptr 500, ic *optocoupler* pc 817, resistor 680 Ohm, resistor 470 Ohm Dioda *clamp*, dan transistor FET IRF 540.

3.2.2.3 Rangkaian Sensor *Optocoupler*

Sensor *octocoupler* adalah sensor yang mampu mendeteksi kecepatan putar suatu piringan *encoder* berdasarkan terhalang atau tidaknya cahaya inframerah yang dikirimkan LED ke *phototransistor* melalui lubang piringan *encoder*. Rangkaian ini dipasang pada rangka motor DC untuk mendeteksi kecepatan putar roda bawah dan balok aluminium yang dipasang sejajar *shaft* roda bebas untuk mendeteksi kecepatan putar roda atas. Rangkaian sensor *optocoupler* yang digunakan terdiri dari sebuah sensor *optocoupler* dengan lebar celah 5 milimeter, *Integrated Circuit (IC)* komparator 393, 2 buah LED, dan 4 buah resistor 10 KOhm beserta 3 buah resistor 1 KOhm. Tegangan *input* agar rangkaian dapat berjalan adalah 5 Volt. Tegangan *output* dari rangkaian ini memiliki 2 jenis yaitu *digital output* dan *analog output* yang sama – sama memiliki rentang 0 – 5 Volt. IC komparator 393 berguna untuk menghasilkan tegangan *output* yang benar – benar pulsa *digital* dengan arus maksimal 15 miliAmpere. Pada saat pengambilan data, rangkaian *driver* ini dihubungkan dengan Arduino UNO di pin *digital* 9 dan 10.

Cara kerja dari rangkaian sensor *octocoupler* adalah ketika rangkaian diberi tegangan 5 Volt, LED mulai mengirimkan cahaya inframerah kepada *phototransistor*. Ketika roda roda bebas dan piringan *encoder* yang terdapat pada satu poros mulai berputar, cahaya dari LED terkadang dapat diterima maupun tidak oleh *phototransistror*. Piringan *encoder* yang digunakan memiliki 55 lubang yang didapatkan dari motor bekas. Pada saat sinyal inframerah tidak dapat diterima *phototransistor* (cahaya mengenai bagian piringan yang tidak berlubang) maka *phototransistor* menjadi *open* sehingga *digital output* dan *analog output* akan mengeluarkan pulsa 5 Volt. Sedangkan pada saat sinyal inframerah dapat diterima *phototransistor* (cahaya mengenai bagian piringan yang berlubang) maka tegangan VCC mengalir ke *phototransistor* sehingga *digital output* dan *analog output* akan mengeluarkan pulsa 0 Volt. Pulsa tegangan *output* rangkaian ini akan diolah oleh Arduino dengan fungsi *pulseIn* sehingga menghasilkan besaran *rotation per minute* (rpm).

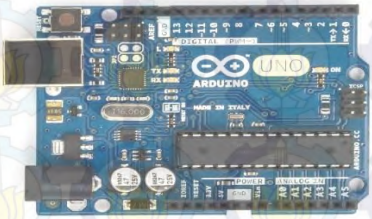


Gambar 3.6 Sensor *Rotary Encoder* yang Terpasang pada *Shaft*

3.2.2.4 Arduino UNO R3

Arduino adalah *platform* pembuatan prototipe elektronik yang bersifat *open-source hardware* yang berdasarkan pada perangkat keras dan perangkat lunak yang fleksibel dan mudah digunakan. Pada tugas akhir ini arduino yang digunakan adalah arduino UNO R3. Arduino ini berfungsi untuk mengolah data hasil sensor *rotary encoder* yang

terpasang pada *shaft* bagian atas, dan bagian bawah. Selain itu arduino ini juga mengeksekusi sinyal kontrol yang diolah oleh komputer. Bentuk fisik arduino UNO dapat dilihat pada Gambar 3.7 dan spesifikasi arduino UNO dapat dilihat pada Tabel 3.3. Arduino menggunakan bahasa pemrograman C yang mudah dipahami, arduino dipilih sebagai perantara komputer dengan *plant*.



Gambar 3.7 Arduino UNO **[6]**

Tabel 3.3 Spesifikasi Arduino UNO R3 **[6]**

Mikrokontroler	ATmega328
Catu daya	5V
Teganan *input* (rekomendasi)	7-12V
Teganan *input* (batasan)	6-20V
Pin I/O *digital*	14 pin dimana 6 diantaranya berupa PWM
Pin *input* Analog	6 pin
Arus DC per pin I/O	40 mA
Flash memory	32 KB (Atmega328)
Clock speed	16 MHz

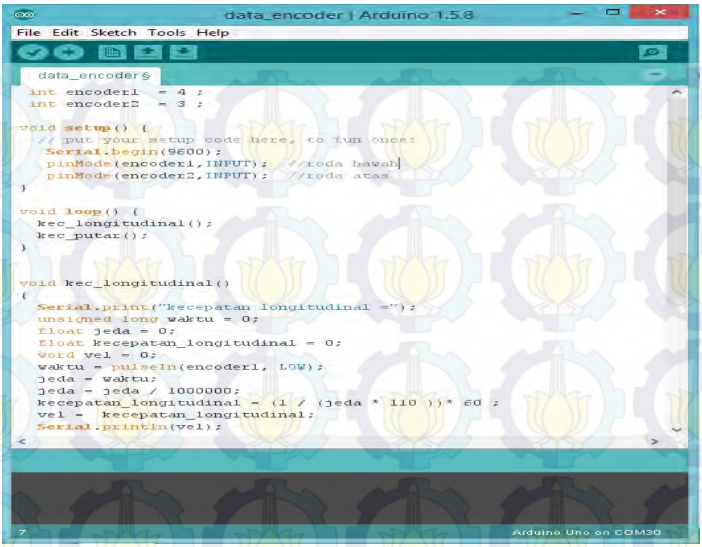
Pada tugas akhir ini, Arduino difungsikan apabila terdapat hasil bacaan dari sensor *rotary encoder* yang terpasang. Selain itu arduino ini difungsikan untuk menghasilan sinyal PWM pada *driver* motor DC dan *driver* rem mikro elektromagnetik. Sinyal PWM pada *driver* motor DC digunakan agar simulator ABS mendapatkan kecepatan awal sebelum melakukan proses pengereman, sedangkan sinyal PWM pada *driver* rem mikro elektromagnetik digunakan sebagai sinyal kontrol kepada simulator. Pengiriman data dari Arduino ke komputer dan sebaliknya dilakukan melalui *Universal Serial Bus* (USB) *serial port.*

3.3 Perancangan *Software*

Peancangan *software* pada tugas akhir ini dibagi menjadi tiga bagian, yaitu; perancangan pembacaan sensor *rotary encoder*, perancangan sinyal PWM untuk motor DC dan rem mikro elektromagnetik, perancangan kontroler dan implementasi. *Software* yang digunakan dalam pengerjaan Tugas Akhir ini antara lain adalah *software* Arduino, CoolTerm dan MATLAB. *Software* Arduino digunakan untuk merancang pembacaan sensor *rotary encoder* dan *sinyal* PWM, sedangkan *software* CoolTerm digunakan meng-*capture* hasil *serial monitor* Arduino dan diubah menjadi *file.txt* sehingga data lebih mudah untuk diolah kemudian data disusun hingga rapi menggunakan *ms.excel*. *Software* MATLAB digunakan untuk keperluan identifikasi sistem dan mengirimkan sinyal kontrol pada saat proses implementasi. Selain itu, MATLAB digunakan juga sebagai *software* untuk desain kontroler yang akan diterapkan pada saat implementasi, dan untuk simulasi kontroler yang akan digunakan pada pengaturan *slip* Tugas Akhir ini.

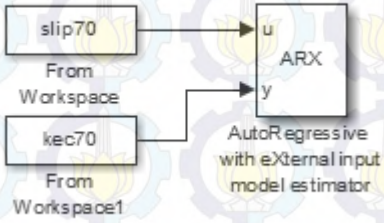
3.3.1 Perancangan *Software* Pembacaan Sensor *Rotary Encoder*

Perancangan *software* ini sangat penting, karena jika pembacaan sensor tidak tepat maka sulit bagi kontroler memberikan sinyal kontrol yang tepat. Oleh karena itu dalam pembacaan sensor *rotary encoder* dibutuhkan ketelitian dan pemrograman yang tepat. Sensor *rotary encoder* yang digunakan pada Tugas Akhir ini memiliki spesifikasi yaitu piringan encoder akan menghasilkan 110 pulsa listrik dengan tegangan 0 – 5 Volt setiap putaran, dan pada saat sinar inframerah mengenai bagian piringan yang tidak berlubang maka sensor akan menghasilkan *output* pulsa *digital* 1, sedangkan jika mengenai bagian yang berlubang sensor akan menghasilkan *output* pulsa *digital* 0. Berdasarkan spesifikasi tersebut maka kita dapat mengubah banyaknya pulsa listrik yang masuk tiap detik ke Arduino menjadi besaran kecepatan putar dengan satuan rpm. Pembacaan dari sensor dibagi menjadi dua bagian, yakni pembacaan sensor pada roda bawah dan pembacaan sensor pada roda atas. Pembacaan dari sensor akan diolah pada komputer sehingga *output* dari MATLAB berupa *slip*. *Output* dari MATLAB yang berupa *slip* akan dibaca oleh arduino sehingga menghasilkan sistem *Closed Loop*. Gambar hasil perrancangan *software* pembacaan sensor *rotary encoder* dapat dilihat pada Gambar 3.8.



Gambar 3.8 Hasil Perancangan *Software* Pembacaan *Sensor Rotary Encoder*

Setelah itu diagram blok Simulink yang dipakai pada saat identifikasi bisa dilihat pada gambar berikut.



Gambar 3.9 Diagram Blok Simulink pada Saat Identifikasi

3.3.2 Perancangan Sinyal PWM untuk Motor DC dan Rem Mikro Elektromagnetik

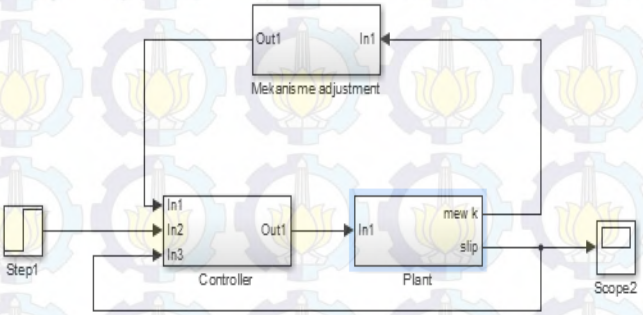
Perancangan sinyal PWM berguna untuk mengatur besarnya sinyal aktuator pada simulator dan mengatur besarnya kecepatan putar awal sebelum simulator melakukan proses pengereman. Pada Tugas Akhir ini, sinyal PWM akan menjadi input bagi rangkaian *driver*. Rangkaian *driver* yang digunakan jika diberi sinyal PWM 10%, maka akan menghasilkan sinyal PWM 90 % kepada motor DC dan rem mikro elektromagnetik. Sinyal PWM motor DC memiliki *state* tegangan 0 Volt dan 24 Volt, sedangkan sinyal PWM rem mikro elektromagnetik memiliki *state* tegangan 0 Volt dan 12 Volt. Gambar hasil perancangan sinyal PWM untuk motor DC dan rem mikro elektromagnetik dapat dilihat pada Gambar 3.10

Gambar 3.10 Hasil Perancangan Sinyal PWM untuk Motor DC dan Rem Mikro Elektromagnetik

3.3.3 Perancangan Kontroler dan Simulasi Sistem

Selain *software* Arduino IDE, *software* lain yang digunakan pada Tugas akhir ini adalah *software* MATLAB R2009b. *Software* ini digunakan untuk perancangan kontroler pada simulator ABS. Untuk merancang kontroler dibutuhkan bahasa pemrograman dan *toolbox*

simulink. Simulink digunakan untuk melakukan proses identifikasi dan pemodelan *open loop* sistem dengan menggunakan *System Identification Toolbox*. Selain itu, Simulink juga digunakan untuk mendesain dan mensimulasikan kontroler pada model simulator ABS. Simulink juga dilengkapi fasilitas komunikasi dengan mikrokontroler Arduino, sehingga proses implementasi kontroler dapat dilakukan melalui *toolbox* Simulink. Blok diagram perancangan dan implementasi sistem dapat dilihat pada Gambar 3.11



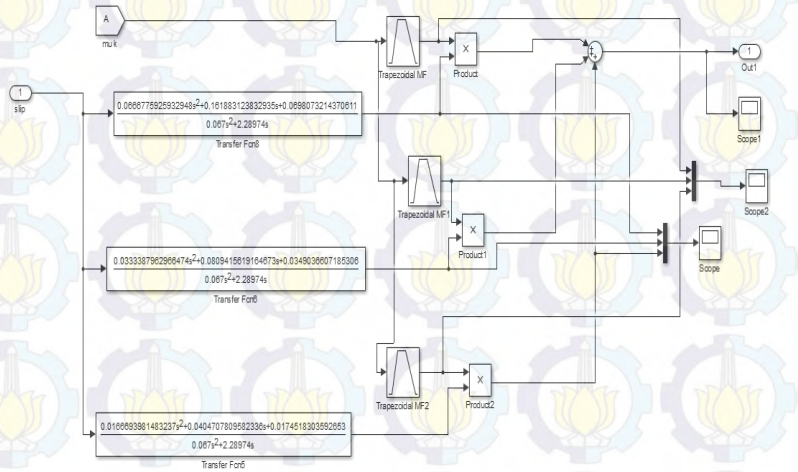
Gambar 3.11 Blok Diagram Simulink Simulasi Kontroler Pengaturan *Slip* pada Simulator ABS

Pada blok diagram simulink dapat dilihat bahwa *output* dari *plant* berupa *slip* dan untuk pengoperasian kondisi berupa koefisien gesek. Besaran dari koefisien gesek akan menentukan kontroler PID yang digunakan.

3.3.3.1 Perancangan *Switching* pada MATLAB

Untuk merancang *switching* dibutuhkan beberapa *input* pada simulink. Simulink digunakan untuk melakukan proses identifikasi dan permodelan *open loop* sistem dengan menggunakan *System Identification Toolbox*. Selain itu, Simulink juga digunakan untuk mendesain dan mensimulasikan kontroler pada model simulator ABS. Dengan menggunakan *membership function fuzzy* pada simulink MATLAB,

penulis dapat merancang pemilihan kontroler ketika kondisi jalan tertentu. Blok diagram pemilihan kontroler dapat dilihat pada Gambar 3.12

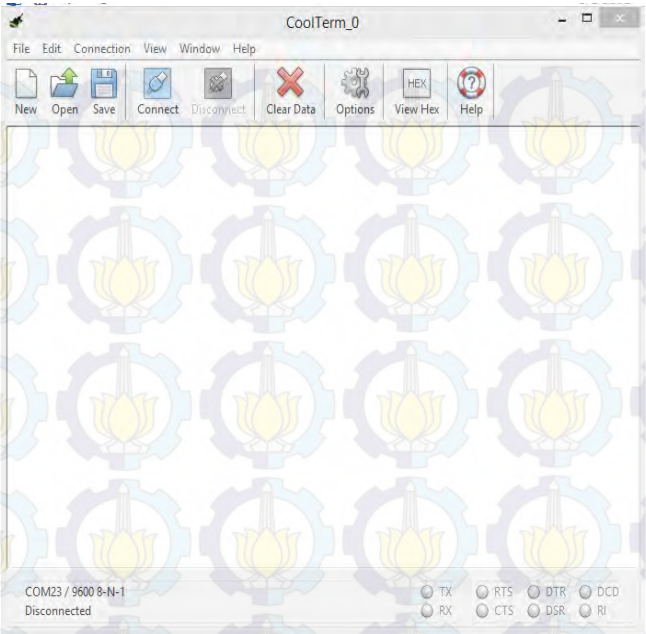


Gambar 3.12 Blok Diagram *Switching* pada Simulink MATLAB

Pada Tugas Akhir ini, perancangan *membership function fuzzy* didasarkan pada kondisi operasi μ_k.

3.3.4 *Software* CoolTerm

Software CoolTerm adalah salah satu aplikasi serial terminal yang membuat komunikasi di dalam mikrokontroler lebih mudah untuk dilihat datanya. Melalui program ini, kita diijinkan untuk melihat data yang dikirimkan dari mikrokontroler dan ke mikrokontroler, sensor, atau perangkat elektronik melalui serial *port* di layar monitor. *Software* CoolTerm pada Tugas Akhir ini digunakan mengubah data kecepatan putar roda dan kecepatan *longitudinal* kendaraan hasil pembacaan sensor *rotary encoder* oleh Arduino yang akan dikirimkan ke komputer dalam bentuk file.txt dengan satuan rpm sehingga data lebih mudah untuk diolah.



Gambar 3.13 Tampilan Utama *Software* CoolTerm

3.4 Proses Identifikasi dan Pemodelan Sistem

Pada Tugas Akhir ini proses identifikasi bertujuan untuk mendapatan model pendekatan simulator ABS berdasarkan *input* dan *output* yang diberikan kepada sistem. Metode pendekatan simulator ABS menggunakan *AutoRegressive with eXternal input model estimator* pada *System Identification Toolbox Simulink*. Model pendeatan sistem yang digunakan adalah model *AutoRegressive* (AR) dengan metode identifikasi parameter *Least Square*.

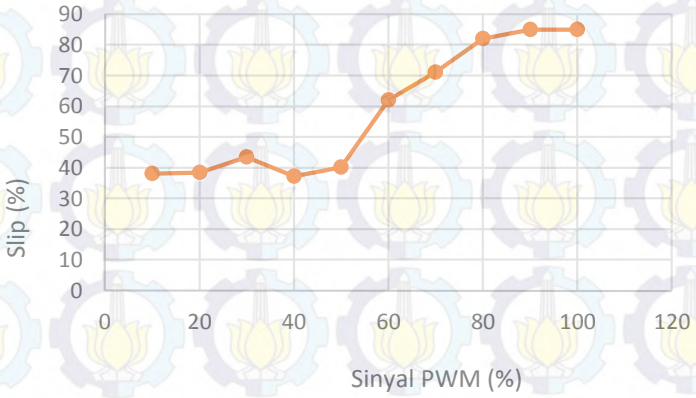
3.4.1 Pemilihan Kecepatan Awal

Sebelum melakukan identifikasi *open loop*, kita harus menentukan *range* kecepatan awal motor sebelum simulator melaukan pengereman. Hubungan kecepatan awal dengan penaikan PWM setiap 10% dapat dilihat pada Tabel 3.4.

Tabel 3.4 Pengaruh Sinyal PWM untuk Penaikan Setiap 10% pada Rem Mikro Elektromagnetik dalam Berbagai Kecepatan Awal Motor DC

Nomor	Kecepatan Awal Sebelum Proses Pengereman (rpm)	Sinyal PWM 10% yang diberikan kepada Motor DC (%)
1	573	10
2	1224	20
3	1621	30
4	1854	40
5	1994	50
6	2097	60
7	2155	70
8	2193	80
9	2220	90
10	2313	100

Pada tugas akhir ini dipilih kecepatan awal pada 2155 rpm, yakni dengan 70% sinyal PWM yang diberikan arduino kepada motor DC.



Gambar 3.14 *Slip* vs Persen Sinyal PWM Rem pada Kecepatan 2155 RPM

Setiap titik kecepatan roda bawah diberikan pwm rem dari 0% hingga 100% (PWM rem ditingkatkan setiap 10%). Saat dilakukan

pengereman maka roda bawah harus sampai berhenti, baru kemudian dapat dijalankan kembali. Data yang diperoleh berupa nilai *slip* pada berberapa range kecepatan roda bawah dan range nilai pwm rem pada roda atas. Tabel 3.5 berikut merupakan hubungan antara perubahan pwm rem pada roda atas dengan titik kerja *slip* untuk 4 titik kecepatan roda bawah yang tertera berikut ini:

Tabel 3.5 Hubungan Pwm Rem dan Titik Kerja *Slip*

	Kecepatan Roda Bawah (RPM)			
Persentase Pwm Rem Roda Atas	2313 (100%)	2220 (90%)	2193 (80%)	2155 (70%)
10%	0,12934	0,08476	0,16076	0,381179
20%	0,14616	0,12543	0,17403	0,38464
30%	0,17498	0,13771	0,21248	0,433369
40%	0,23304	0,24226	0,28326	0,372278
50%	0,48866	0,50346	0,44012	0,400465
60%	0,66523	0,68928	0,66226	0,621829
70%	0,76070	0,78347	0,74383	0,710797
80%	0,82027	0,80893	0,79062	0,829358
90%	0,81499	0,84319	0,87207	0,852933
100%	0,83794	0,86017	0,86869	0,854082

Berdasarkan Tabel 3.5 diatas, titik kerja *slip* merupakan rata-rata *slip* yang didapat pada satu titik kerja rem untuk satu titik kecepatan roda bawah. Data yang diolah adalah data *slip* pada kecepatan roda bawah dan kecepatan roda atas saat mulai mengalami perlambatan. Pada nilai pwm rem tinggi roda bagian atas sempat mengunci (tidak berputar) hingga roda bagian bawah berhenti berputar Dari data yang diperoleh dapat ditentukan titik kerja rem terhadap *slip* untuk masing-masing kecepatan roda bawah seperti yang dicantumkan pada Tabel 3.5. Dari data pada Tabel 3.5 terlihat bahwa titik kerja *slip* bernilai lebih dari 0,2 saat persentase pwm rem diatas 40% untuk 3 titik kecepatan roda bawah, dan *slip* bernilai lebih dari 0,2 saat persentase pwm rem diatas 10% untuk 1 titik kecepatan roda bawah. Dalam tugas akhir ini, penulis memilih kecepatan 70% (maksimum) untuk kecepatan roda bawah. *plot* grafik hubungan titik kerja rem terhadap *slip* pada kecepatan roda bawah 70% pwm dapat dilihat pada Gambar 3.14.

3.4.2 Identifikasi *Open loop* Simulator ABS pada Kecepatan Motor 2155

Pada identifikasi *open loop* diberikan sinyal uji *random* agar didapatkan model pendekatan AR dengan metode *Least Square.* Identifikasi sisterm dilakukan pada kecepatan 2155 rpm dengan sinyal *input* rem yang bervariasi mulai dari 10%, 20%, 30%, 40%, 50%, hingga 100%. Hubungan *input output* simulator ABS dapat dilihat pada Tabel 3.6 dan Gambar 3.13.

Tabel 3.6 *Slip* Vs Sinyal PWM (%) pada kecepatan 2155

No	Sinyal PWM (%) pada kecepatan 2155	*Slip* roda atas terhadap roda bawah
1	10	0,381179
2	20	0,38464
3	30	0,433369
4	40	0,372278
5	50	0,400465
6	60	0,621829
7	70	0,710797
8	80	0,829358
9	90	0,852933
10	100	0,854082

3.4.3 Pemodelan dan Validasi Simulator ABS dengan Kecepatan Awal 2155 RPM

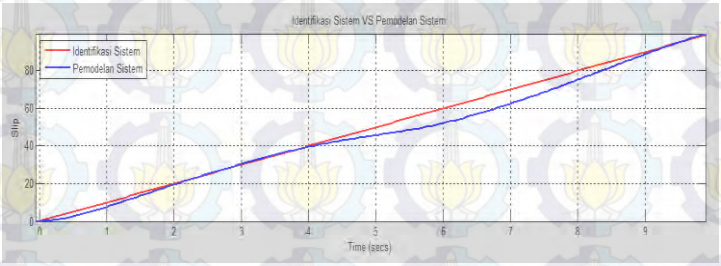
Setelah mendapatkan hubungan *input output* simulator ABS melalui identifikasi *open loop* pada kecepatan awal 2155 rpm, selanjutnya adalah memodelkan hubungan yang ada ke dalam representasi diskrit sistem deterministik model AR. Pada Tugas Akhir ini, digunakan metode *root mean square error* (RMSE) sebagai validasi model yang didapatkan dari pemodelan *toolbox* Simulink dengan hubungan *input* output simulator ABS. Berdasarkan Tabel 3.7 didapatkan model sistem deterministik orde 2 dengan nilai RMSE yang kecil. Orde 3 memiliki RMSE lebih kecil dibandingkan orde 2, Meskipun estimasi *error* yang dihasilkan lebih kecil bila dibandingkan estimasi *error* model orde 2, akan tetapi model orde 2 sudah merepresentasikan dinamika sebuah *plant* Oleh karena itu pendekatan model *plant* yang digunakan adalah model *plant* orde 2. Model orde 2 yang didapatkan merepresentasikan model *plant*

ABS. Model *plant* yang didapatkan sudah dalam bentuk diskrit dengan domain z:

$$G(z) = \frac{0{,}55534\,z}{z^2 - 1{,}8582z + 0{,}8617} \qquad (3.1)$$

Tabel 3.7 Model Sistem Deterministik dengan RMSE

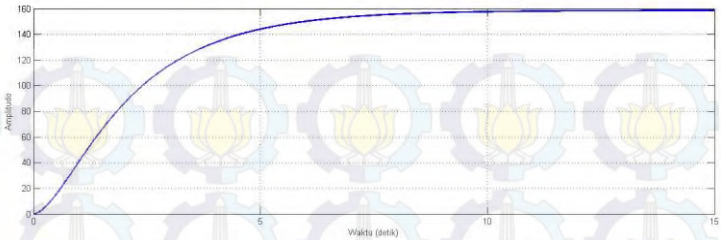
Orde	Fungsi Alih	RMSE (%)
1	$\frac{14{,}705}{z - 0{,}99289}$	0,554
2	$\frac{0{,}55534\,z}{z^2 - 1{,}8582z + 0{,}8617}$	0,124
3	$\frac{1{,}6345x10^{-15}\,z^2}{z^3 - 2z^2 + z - 1{,}6345x10^{-12}}$	0,000217



Gambar 3.15 Identifikasi Sistem VS Pemodelan Sistem

3.5 Perancangan Kontroler PID Adaptif

Sebelum melakukan perancangan kontroler, terlebih dahulu kita harus mengetahui respon *open loop* dari model *plant* yang telah didapatkan pada proses sebelumnya. Respon *open loop* ini didapatkan dengan menggunakan *software* MATLAB. Respon *open loop* simulator ABS dapat dilihat pada Gambar 3.16, dimana *input* yang diberikan adalah sinyal *step*. Blok diagram simulink kontroler adaptif dapat dilihat pada Gambar 3.11.



Gambar 3.16 Respon *Open Loop* Sistem

3.5.1 Perancangan Kontrol Adaptif

Pada tugas akhir ini dirancang kontrol adaptif pada saat pengereman sehingga *slip* antara mobil dan jalan mencapai 20%. Hal ini bertujuan untuk membantu meningkatkan kemampuan meminimalisasi jarak pemberhentian kendaraan saat terjadi pengereman mendadak. Kontrol adaptif yang digunakan berupa *gain scheduling*, sehingga memerlukan kontroler PID lebih dari 1 untuk beberapa kondisi operasi. Kondisi operasi didasarkan pada koefisien gesek antara ban mobil dan jalan dapat dilihat pada Tabel 3.8:

Tabel 3.8 Beberapa Nilai Koefisien Gesek antara Ban Mobil dan Jalan

μ_k(Koefisien gesek)	Kondisi Jalan
$\mu_k <= 0{,}2$	Kondisi jalan licin
$\mu_k > 0{,}2$ hingga <=0,4	Kondisi jalan basah
$\mu_k > 0{,}4$	Kondisi jalan kering

Kondisi operasi ini yang akan menentukan pemakaian kontroler PID yang sesuai dengan kondisi jalan. Penentuan parameter PID didasarkan pada metode pendekatan dengan hitungan analitik . Karena kondisi operasi ada 3 maka kontroler yang diperrlukan ada 3 juga sehingga masing-masing kondisi memerlukan 1 kontroler. Untuk itu dirancanglah kontroler PID pada setiap kondisi.

3.5.1.1 Perancangan Mekanisme *Adjustment*

Perancangan Mekanisme *Adjustment* didasarkan kepada hasil perhitungan μ_k. Perancangan $\mu_k$didasarkan pada persamaan gaya gerak mobil yang dapat dilihat pada Persamaan berikut:

$$m\dot{v} = -F_x \tag{3.2}$$

$$F_x = -F_z \,.\, \mu_k \quad (3.3)$$

Dengan mensubtitusi F_x,maka didapatkan:

$$m\dot{v} = F_z \,.\, \mu_k \quad (3.4)$$

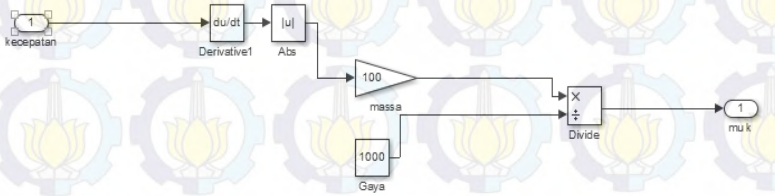
$$\mu_k = \frac{m\dot{v}}{F_z} \quad (3.5)$$

$$\mu_k = \frac{m\dot{v}}{m.g} \quad (3.6)$$

Sehingga dapat disederhanakan menjadi

$$\mu_k = \frac{\dot{v}}{g} \quad (3.7)$$

Blok diagram simulink mekanisme *adjustment* dapat dilihat pada Gambar 3.17.



Gambar 3.17 Blok Diagram Simulink Mekanisme *Adjustment*

Jadi, $\mu_k$adalah massa dikalian percepatan mobil dibagi dengan gaya normalnya. Gaya normal adalah massa dikalikan gravitasinya dikarenakan jalan raya diasumsikan datar, kemudian dari hasil tersebut didapatkan nilai μ_k yang sesuai dengan kondisi operasi.

Perancangan tiga kontroler PID didasarkan pada koefisien gesek antara ban mobil dengan kondisi jalan. Pembagian titik kerja kontroler dapat dilihat pada Tabel 3.9.

Tabel 3.9 Pembagian Titik Kerja Kontroler PID

Kontroler PID yang Digunakan	Besaran μ_k
PID1	$\mu_k <= 0{,}2$
PID2	$\mu_k > 0{,}2$ hingga <=0,4
PID3	$\mu_k > 0{,}4$

3.5.2 Perancangan Kontroler PID

Perancangan suatu kontroler PID pada dasarnya menentukan parameter K_p, τ_i, τ_d sedemikian rupa sehingga respon sistem hasil desain sesuai dengan spesifikasi performansi yang diinginkan. Pada Tugas Akhir ini spesifikasi respon *output* yang ingin dicapai adalah memenuhi respon orde I dengan *overshoot*=0%, *error steady state* =0%, dengan *time constant*= 1 detik. Pemilihan parameter *time constant* sebesar 1 detik untuk mempercepat respon *plant*. Untuk parameter PID dapat digunakan pada Persamaan 3.8, Persamaan 3.9, dan Persamaan 3.10.

$$K_p = \frac{\tau_i}{\tau^* K} \tag{3.8}$$

$$\tau_d = \tau \tag{3.9}$$

$$\tau_i = \frac{2\zeta}{\omega_p} - \tau \tag{3.10}$$

3.5.2.1 Kontroler PID1

Perancangan PID1 didasarkan pada respon sistem dengan $\mu_k \leq 0{,}2$. Dikarenakan pada Tugas Akhir ini kontroler bertujuan untuk memenuhi respon *overshoot*=0%, dan *error steady state=0%* dari respon *closed loop* maka dirancanglah modifikasi kontroler PID dengan *delay* pada *plant* orde II sehingga dapat menghasilkan respon sistem orde I.

Transfer function plant :

$$Gp = \frac{0{,}972\ s + 33{,}22}{0{,}955s^2 + 2{,}319s + 1} \tag{3.11}$$

$$Gp = \frac{K(s+1)}{\frac{1}{{\omega_p}^2}s^2 + \frac{2s}{\omega_p} + 1} \tag{3.12}$$

Sehingga parameter *plant* dapat dilihat pada Tabel 3.10:

Tabel 3.10 Parameter *Plant* ($\boldsymbol{\mu_k \leq 0,2}$)

K	33,22
τ	0,02926
ω_p	1,0232
ζ	1,1864

Dari parameter *plant* maka dirancang kontroler PID1 dengan *delay* sebagai berikut:

$$\tau_d = \tau = 0{,}02926 \tag{3.13}$$

$$\tau_i = \frac{2\zeta}{\omega_p} - \tau = 2{,}28974 \tag{3.14}$$

$$K_p = \frac{\tau_i}{\tau^* K} = 0{,}0698 \tag{3.15}$$

$$N = \frac{1}{\tau\omega_p(2\zeta - \tau\omega_p)} - 1 = 13{,}2567 \tag{3.16}$$

Sehingga *transfer function* kontrolernya adalah:

$$Gc = K_p \frac{[(N+1)\tau_i\tau_d s^2 + (\tau_i + \tau_d)s + 1]}{\tau_i s(\tau_d s + 1)} \tag{3.17}$$

$$Gc = \frac{0{,}0667\, s^2 + 0{,}161\, s + 0{,}0698}{0{,}067\, s^2 + 2.28974\, s} \tag{3.18}$$

Closed loop transfer function:

$$\frac{Y(s)}{X(s)} = \frac{0{,}0648s^3 + 2{,}372s^2 + 5{,}4456\, s + 2{,}319}{0{,}067s^4 + 2{,}5208s^3 + 8{,}1234s^2 + 7{,}843s + 2{,}319} \tag{3.19}$$

3.5.2.2 Kontroler PID2

Perancangan PID2 didasarkan pada respon sistem dengan $0{,}2 < \mu_k \leq 0{,}4$. Dikarenakan pada Tugas Akhir ini kontroler bertujuan untuk memenuhi respon *overshoot*=0%, dan *error steady state=0%* dari respon *closed loop* maka dirancanglah modifikasi kontroler PID dengan *delay* pada *plant* orde II sehingga dapat menghasilkan respon sistem orde I.. Perhitungan analitik untuk kontroler PID adalah sebagai berikut:
Transfer function plant :

$$Gp = \frac{1{,}944\, s + 66{,}44}{0{,}955s^2 + 2{,}319s + 1} \tag{3.20}$$

$$Gp = \frac{K(s+1)}{\frac{1}{{\omega_p}^2}s^2 + \frac{2s}{\omega_p} + 1} \quad (3.21)$$

Sehingga parameter *plant* dapat dilihat pada Tabel 3.11:

Tabel 3.11 Parameter *Plant* ($\mathbf{0,2 < \mu_k \leq 0,4}$)

K	66,44
τ	0,02926
ω_p	1,0232
ζ	1,1864

Dari parameter *plant* maka dirancang kontroler PID2 dengan *delay* sebagai berikut:

$$\tau_d = \tau = 0{,}02926 \quad (3.22)$$

$$\tau_i = \frac{2\zeta}{\omega_p} - \tau = 2{,}28974 \quad (3.23)$$

$$K_p = \frac{\tau_i}{\tau^* K} = 0{,}0349 \quad (3.24)$$

$$N = \frac{1}{\tau\omega_p(2\zeta - \tau\omega_p)} - 1 = 13{,}2567 \quad (3.25)$$

Sehingga *transfer function* kontrolernya adalah:

$$Gc = K_p \frac{[(N+1)\tau_i\tau_d s^2 + (\tau_i + \tau_d)s + 1]}{\tau_i s(\tau_d s + 1)} \quad (3.26)$$

$$Gc = \frac{0{,}0333\, s^2 + 0{,}08094\, s + 0{,}03490}{0{,}067\, s^2 + 2.28974\, s} \quad (3.27)$$

Closed loop transfer function:

$$\frac{Y(s)}{X(s)} = \frac{0{,}0648s^3 + 2{,}372s^2 + 5{,}4456\,s + 2{,}319}{0{,}067s^4 + 2{,}5208s^3 + 8{,}1234s^2 + 7{,}843s + 2{,}319} \quad (3.28)$$

3.5.2.3 Kontroler PID3

Perancangan PID3 didasarkan pada respon sistem dengan $\mu_k >$ 0,4. Dikarenakan pada Tugas Akhir ini kontroler bertujuan untuk memenuhi respon *overshoot*=0%, dan *error steady state=0%* dari respon *closed loop* maka dirancanglah modifikasi kontroler PID dengan *delay* pada *plant* orde II sehingga dapat menghasilkan respon sistem orde I. Perhitungan analitik untuk kontroler PID adalah sebagai berikut:
Transfer function plant :

$$Gp = \frac{3{,}888\,s + 132{,}9}{0{,}955s^2 + 2{,}319s + 1} \quad (3.29)$$

$$Gp = \frac{K(s+1)}{\frac{1}{{\omega_p}^2}s^2 + \frac{2s}{\omega_p} + 1} \quad (3.30)$$

Sehingga parameter *plant* dapat dilihat pada Tabel 3.12

Tabel 3.12 Parameter *Plant* ($\boldsymbol{\mu_k > 0,4}$)

K	132,9
τ	0,02926
ω_p	1,0232
ζ	1,1864

Dari parameter *plant* maka dirancang kontroler PID3 dengan *delay* sebagai berikut:

$$\tau_d = \tau = 0{,}02926 \quad (3.31)$$

$$\tau_i = \frac{2\zeta}{\omega_p} - \tau = 2{,}28974 \quad (3.32)$$

$$K_p = \frac{\tau_i}{\tau^* K} = 0{,}0175 \quad (3.33)$$

$$N = \frac{1}{\tau\omega_p(2\zeta - \tau\omega_p)} - 1 = 13{,}2567 \quad (3.34)$$

Sehingga *transfer function* kontrolernya adalah:

$$Gc = K_p \frac{[(N+1)\tau_i\tau_d s^2 + (\tau_i+\tau_d)s + 1]}{\tau_i s(\tau_d s + 1)} \quad (3.35)$$

$$Gc = \frac{0{,}01667\, s^2 + 0{,}04047\, s + 0{,}01745}{0{,}067\, s^2 + 2.28974\, s} \quad (3.36)$$

Closed loop transfer function:

$$\frac{Y(s)}{X(s)} = \frac{0{,}0648s^3 + 2{,}372s^2 + 5{,}4456\, s + 2{,}319}{0{,}067s^4 + 2{,}5208s^3 + 8{,}1234s^2 + 7{,}843s + 2{,}319} \quad (3.37)$$

Besaran parameter K_p, K_i, K_d pada setiap kondisi dapat dilihat pada Tabel 3.13.

Tabel 3.13 Besaran parameter $\mathbf{K_p}$, $\mathbf{K_i}$, $\mathbf{K_d}$ pada setiap kondisi $\boldsymbol{\mu_k}$

	$\mu_k \leq 0{,}2$	$0{,}2 < \mu_k \leq 0{,}4$	$\mu_k > 0{,}4$
K_p	0,14	0,0698	0,0349
K_i	0,0602	0,301	0,0151
K_d	0,0575	0,0288	0,0144

BAB 4
PENGUJIAN DAN ANALISA

Pada bab ini akan dibahas mengenai Simulasi kontroler PID adaptif untuk mempertahankan rasio *slip* yaitu sekitar 20% untuk setiap masing-masing kondisi jalan. Pemberian nilai koefisien gesek didasarkan kepada persamaan gaya gerak mobil yang dapat dilihat Bab 3.

4.1 Hasil Pengujian Simulasi

Simulasi ini bertujuan untuk mengetahui apakah respon sistem sudah baik atau belum. Hasil simulasi kontroler dapat dilihat pada BAB 3.

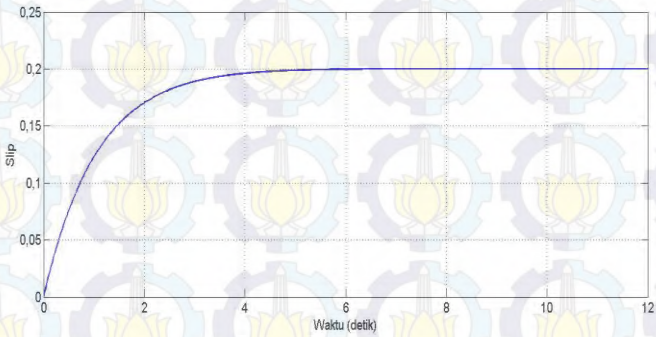
4.1.1 Pengujian Simulasi Model *Plant* ABS

Pada tahapan ini ABS yang telah dimodelkan akan dicoba disimulasikan. Pengujian ini dilakukan dengan memberikan respon *step* di semua kondisi operasi jalan mulai dari jalan licin, jalan basah, dan jalan kering. Setelah itu respon akan dianalisa untuk dicari nilai *rise time*, *settling time*, *overshoot*, dan *error steady state*.

Setalah dilakukan simulasi akan dilakukan analisa terhadap respon hasil dari kontroler. Yang dilakukan selanjutnya adalah menghitung *settling time*, *rise time*, *overshoot* dan *error steady state.* karena kontroler yang digunakan adalah kontroler PID maka yang perlu dihitung *error steady state, settling time*, *rise time*, dan *overshoot.*

Untuk menghitung nilai *settling time* yang perlu dilakukan hanyalah menghitung nilai di mana respon *output* telah memasuki zona ±0,5% dari nilai *steady state* respon. Lalu untuk *rise time,* yang perlu dilakukan adalah menghitung waktu yang dibutuhkan oleh respon dari mulai 10% dari nilai *steady state* hingga ke 90% dari nilai *steady state.* Dan untuk nilai *overshoot* yang perlu dilakukan adalah menghitung nilai nilai maksimum respon lalu dikurangi nilai *steady state* respon lalu dibagi dengan nilai *steady state* respon.

4.1.1.1 Simulasi pada Kondisi Jalan Licin ($\mu_k \leq 0,2$)



Gambar 4.1 Respon *Plant* dengan Kontroler PID pada Kondisi $\mu_k \leq 0,2$

Di bawah ini merupakan hasil perhitungan dan pengamatan *rise time* (T_r) dan *setlling time* (T_s) dari respon sistem.

Slip steady state plant ABS adalah 0,2 dengan *input step*, sehingga untuk mencari T_s yang perlu dicari adalah waktu dari 0 hingga titik di mana *slip* mencapai 95% *slip steady state* yang nilainya yaitu 3,102.

Dan untuk mencari nilai T_r yang perlu dicari adalah waktu di mana *slip* mencapai 90% dari *slip steady state* untuk pertama kali dikurangi dengan waktu di mana *slip* mencapai 10% dari *slip steady state*. Dimana nilai waktu *slip* saat 10% dari *slip steady state* adalah 0,112 dan waktu *slip* saat 90% dari *slip steady state* adalah 2,413. Oleh karena itu, dari nilai-nilai ini hanya perlu dicari waktu pada saat titik-titik nilai tersebut.

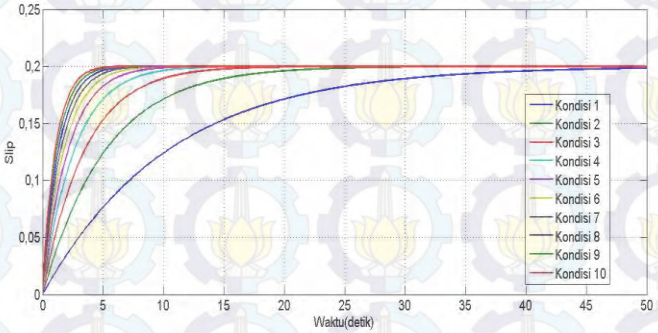
$$T_s = 3,10\ detik$$
$$T_r = 2,301\ detik$$

4.2 Analisa Hasil Pengujian pada Setiap Kondisi Jalan

Pada proses ini akan dilakukan analisa disetiap kondisi jalan, untuk $\mu_k \leq 0,2$ adalah kondisi jalan yang licin, $0,2 < \mu_k \leq 0,4$ adalah kondisi jalan basah, dan $\mu_k > 0,4$ adalah kondisi jalan kering

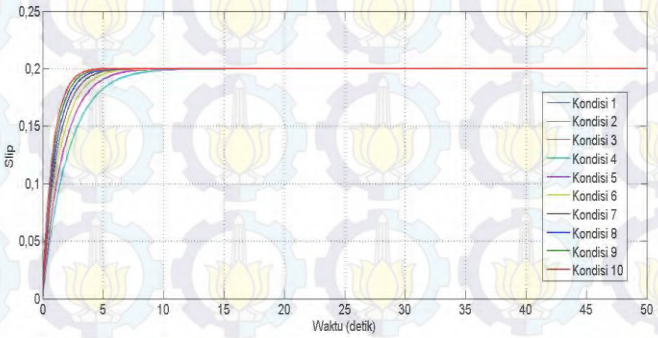
Pada Tugas Akhir ini, tujuan dari kontroler adalah membuat respon *slip plant* menyerupai respon orde I pada setiap kondisi jalan. Nilai

overshoot dan *error steady state* harus bernilai 0%. Pada simulasi pengereman, nilai μ_k menyatakan koefisien gesek antara roda dengan jalan. Nilai μ_k diubah-ubah untuk menyatakan kondisi jalan yang dilalui kendaraan.



Gambar 4.2 Respon *Output* Sistem dengan Kontroler PID Biasa

Dari Gambar 4.2 dapat dilihat bahwa respon *output* mengikuti *set point* sebesar 0,2 Respon *output* sistem tidak memiliki *overshoot* dan *errror steady state*. Kondisi 1 menyatakan μ_k=0,1 , kondisi 2 menyatakan μ_k=0,2 , kondisi 3 menyatakan μ_k=0,3 , kondisi 4 menyatakan μ_k=0,4 , kondisi 5 menyatakan μ_k=0,5 , kondisi 6 menyatakan μ_k=0,6 , kondisi 7 menyatakan μ_k=0,7 , kondisi 8 menyatakan μ_k=0,8 , kondisi 9 menyatakan μ_k=0,9 , kondisi 10 menyatakan μ_k=10.



Gambar 4.3 Respon *Output* Sistem dengan PID *Gain Scheduling*

Tabel 4.1 Spesifikasi Respon *Output* Sistem dengan Kontroler PID Biasa

Kondisi Operasi	*Rise time*	*Time Settling*	*Overshoot PID*	*Error steady state PID*
$\mu_k = 0{,}1$	28,56	39	0%	0%
$\mu_k = 0{,}2$	13,84	18,9	0%	0%
$\mu_k = 0{,}3$	9,115	12,5	0%	0%
$\mu_k = 0{,}4$	6,848	9,35	0%	0%
$\mu_k = 0{,}5$	5,461	7,457	0%	0%
$\mu_k = 0{,}6$	4,554	6,22	0%	0%
$\mu_k = 0{,}7$	3,907	5,33	0%	0%
$\mu_k = 0{,}8$	3,420	4,67	0%	0%
$\mu_k = 0{,}9$	3,047	4,16	0%	0%
$\mu_k = 1$	2,71	3,17	0%	0%

Sehingga dari Tabel 4.1 dapat disimpulkan bahwa kontroler PID pada Sistem ABS mampu mempercepat respon. Respon *output* sistem tidak memiliki *overshoot* dan *errror steady state.* Semakin besar nilai μ_k maka nilai *rise time* dan *time settling* semakin kecil. Sehingga dibutuhkan PID *gain scheduling* untuk mempercepat respon sistem pada saat pengereman.

Tabel 4.2 Spesifikasi Respon *Output* Sistem dengan PID *Gain Scheduling*

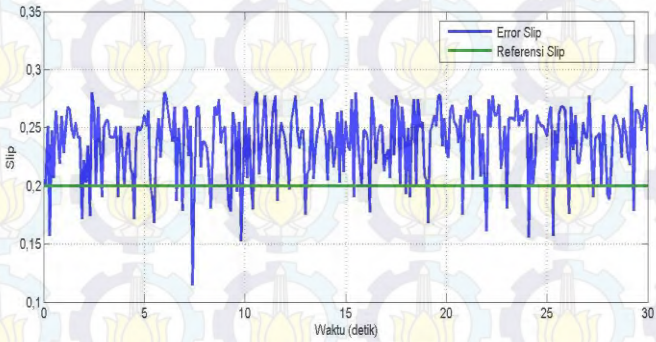
Kondisi Operasi	*Rise time*	*Time Settling*	*Overshoot PID*	*Error steady state PID*
$\mu_k = 0{,}1$	5,471	7,47	0%	0%
$\mu_k = 0{,}2$	5,42	7,4	0%	0%
$\mu_k = 0{,}3$	5,35	7,35	0%	0%
$\mu_k = 0{,}4$	4,36	7,3	0%	0%
$\mu_k = 0{,}5$	4,36	5,95	0%	0%
$\mu_k = 0{,}6$	3,65	4,98	0%	0%
$\mu_k = 0{,}7$	3,217	4,27	0%	0%
$\mu_k = 0{,}8$	2,747	3,75	0%	0%
$\mu_k = 0{,}9$	2,453	3,35	0%	0%
$\mu_k = 1$	2,212	3,02	0%	0%

4.3 Implementasi Sistem

Implementasi dilakukan melalui perangkat lunak matlab yang dihubungkan dengan *real plant* ABS. Implementasi sistem bertujuan untuk mengetahui data respon *transient* sistem saat dihbungkan dengan *real plant* ABS dan membandingkan hasil respon sistem yang didapat dengan hasil respon sistem saat simulasi. Implementasi dilakukan menggunakan dua arduino untuk mengatur kecepatan putar roda bawah yang terhubung dengan motor DC dan sebagai *interface* kontroler dengan *plant*.

Program pada arduino dibuat agar dapat berkomunikasi serial secara *half duplex* dengan matlab agar data kecepatan (kecepatan *longitudinal*

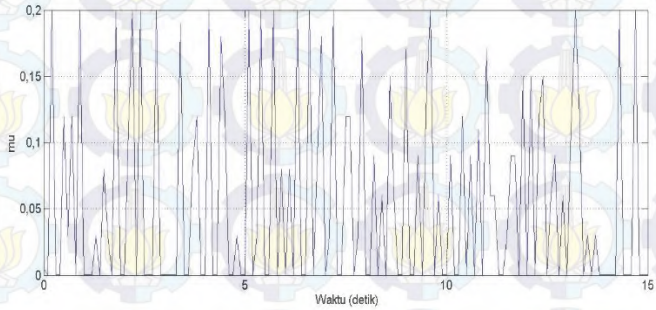
dan kecepatan putar) dan sinyal kontrol dapat dilewatkan secara bergantian oleh arduino. Berikut hasil implementasi dari *plant* ABS:



Gambar 4.4 Perbandingan Respon Sinyal *Error* ABS dengan Referensi *Slip*

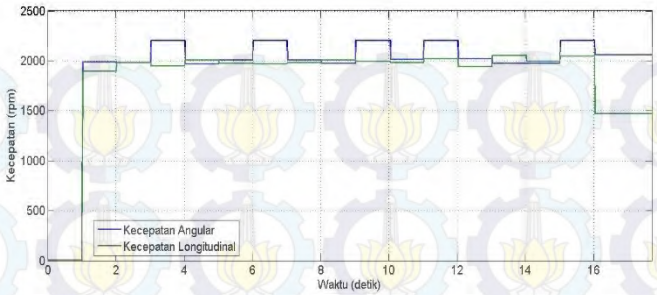
Berdasarkan respon sistem ABS pada Gambar 4.4 menunjukkan bahwa kontroler PID dapat bekerja pada setiap kondisi operasi. Nilai *error slip* yang dihasilkan masih dalam *range* yang diperbolehkan yakni antara 0,10-0,30.

4.3.1 Implementasi pada Saat Kondisi Jalan Licin



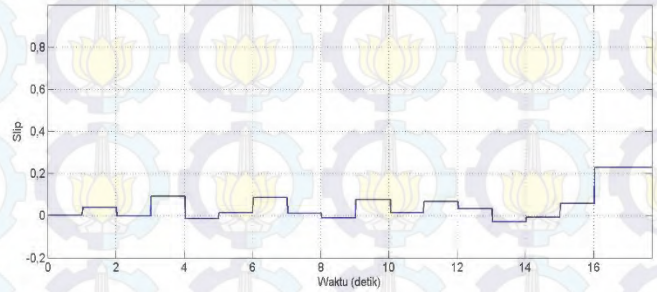
Gambar 4.5 Besar μ_k pada Saat Kondisi Jalan Licin

Besarnya μ_k pada saat implementasi berkisar antara 0 hingga 0,2. Keragaman μ_k diakibatkan oleh motor atas yang dijalankan pada saat implementasi.



Gambar 4.6 Perbandingan Kecepatan Putar dan Longitudinal ($\mu_k \leq 0,2$)

Dapat dilihat pada Gambar 4.6 waktu yang dibutuhkan agar mobil dapat benar-benar berhenti adalah sebesar 1,7detik.

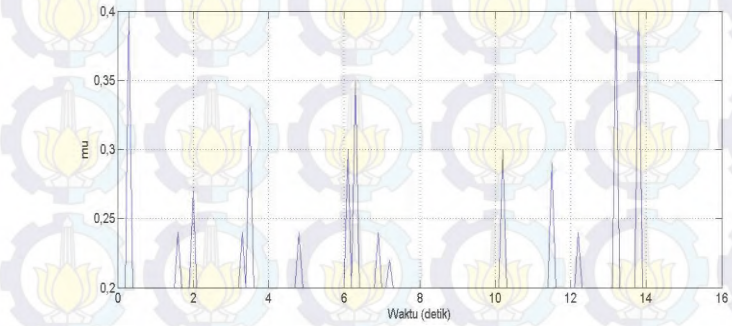


Gambar 4.7 *Output Plant* ABS ($\mu_k \leq 0,2$)

Berdasarkan respon sistem ABS pada Gambar 4.7 menunjukkan bahwa kontroler PID dapat bekerja pada saat dilakukan pengereman. Pada saat pengereman nilai *slip* dapat mengikuti nilai sinyal referensi sebesar 0,2 dan dapat menjaga roda atas tidak mengunci atau menjaga agar *slip* tidak bernilai 1. Nilai *error slip* yang dihasilkan masih dalam *range* yang diperbolehkan yakni antara 0,10-0,30. Respon Sistem pada saat roda atas dan roda bawah sudah berhenti menuju ke nilai *slip* 1 (batas atas saturasi) karena pengaruh dari rumus *slip* yang bernilai 0/0 saat roda atas dan roda bawah sudah berhenti. Saat dilakukan pengereman ada perubahan kecepatan *longitudinal* dan kecepatan putar sebagai efek dari nilai *slip*

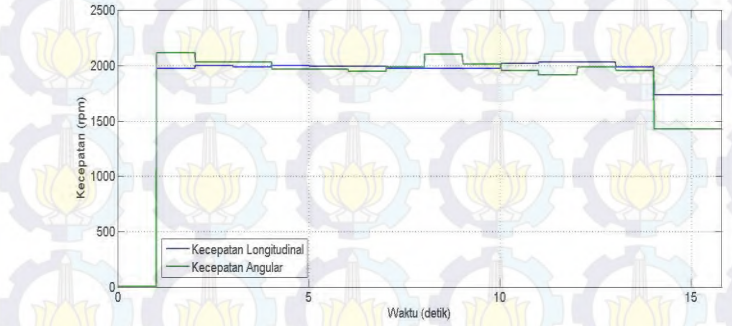
yang dapat dikendalikan. Ketika terjadi pengereman, kontroler masih mampu mempertahankan *slip* sebesar 0,21.

4.3.2 Implementasi pada Saat Kondisi Jalan Basah



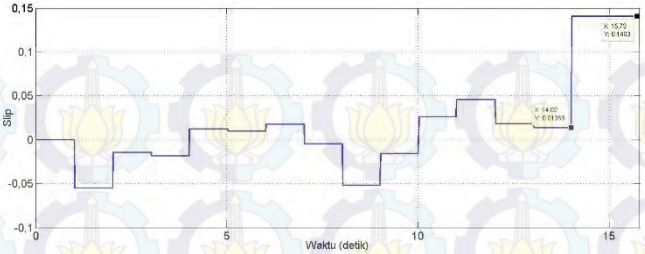
Gambar 4.8 Besar μ_k pada saat Kondisi Jalan Basah

Pada Gambar 4.8 besarnya μ_k pada saat implementasi berkisar antara 0,2 hingga 0,4. Keragaman diakibatkan oleh putaran motor atas yang digunakan untuk menghasilkan besaran μ_k.



Gambar 4.9 Perbandingan Kecepatan Putar dan Longitudinal ($\mathbf{0,2 < \mu_k \leq 0,4}$)

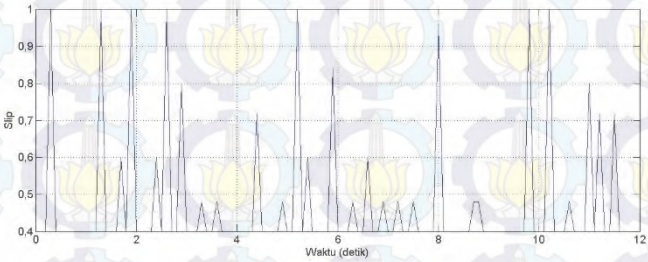
Dapat dilihat pada Gambar 4.9 waktu yang dibutuhkan agar mobil dapat benar-benar berhenti adalah sebesar 1,71 detik.



Gambar 4.10 *Output Plant* ABS ($\mathbf{0,2 < \mu_k \leq 0,4}$)

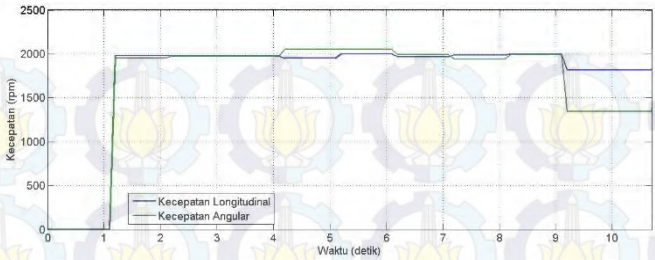
Berdasarkan respon sistem ABS pada Gambar 4.16 menunjukkan bahwa kontroler PID dapat bekerja pada saat dilakukan pengereman. Pada saat pengereman nilai *slip* dapat mengikuti nilai sinyal referensi sebesar 0,2 dan dapat menjaga roda atas tidak mengunci atau menjaga agar *slip* tidak bernilai 1. Nilai *error slip* yang dihasilkan masih dalam *range* yang diperbolehkan yakni antara 0,10-0,30. Respon Sistem pada saat roda atas dan roda bawah sudah berhenti menuju ke nilai *slip* 1 (batas atas saturasi) karena pengaruh dari rumus *slip* yang bernilai 0/0 saat roda atas dan roda bawah sudah berhenti. Saat dilakukan pengereman ada perubahan kecepatan *longitudinal* dan kecepatan putar sebagai efek dari nilai *slip* yang dapat dikendalikan. Ketika terjadi pengereman, kontroler mampu mempertahankan *slip* sebesar 0,14.

4.3.3 Implementasi pada Saat Kondisi Jalan Kering



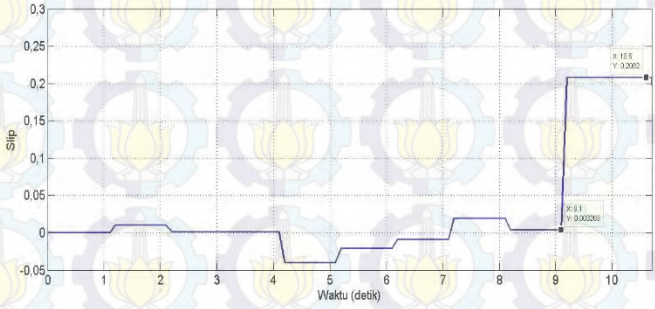
Gambar 4.11 Besar μ_k pada saat Kondisi Jalan Kering

Pada Gambar 4.17 besarnya μ_k pada saat implementasi sebesar > 0,4. Keragaman diakibatkan oleh putaran motor atas yang digunakan untuk menghasilkan besaran μ_k.



Gambar 4.12 Perbandingan Kecepatan Putar dan Longitudinal ($\mu_k > 0,4$)

Pada Gambar 4.18 dapat dilihat bahwa pada saat terjadi pengereman ban mobil sudah berhenti namun kecepatan longitudinal masih berjalan beberapa saat. Selisih waktu antara kecepatan longitudinal dan kecepatan angular mobil pada saat pengereman sekitar 1,4 detik. Artinya bahwa mobil benar benar berhenti setelah 1,4 detik pengereman.



Gambar 4.13 *Output Plant* ABS ($\mu_k > 0,4$)

Berdasarkan respon sistem ABS pada Gambar 4.19 menunjukkan bahwa kontroler PID dapat bekerja pada saat dilakukan pengereman. Pada saat pengereman nilai *slip* dapat mengikuti nilai sinyal referensi sebesar 0,2 dan dapat menjaga roda atas tidak mengunci atau menjaga agar *slip* tidak bernilai 1. Nilai *error slip* yang dihasilkan masih dalam *range* yang diperbolehkan yakni antara 0,10-0,30. Respon Sistem pada saat roda atas dan roda bawah sudah berhenti menuju ke nilai *slip* 1 (batas atas saturasi) karena pengaruh dari rumus *slip* yang bernilai 0/0 saat roda atas dan roda bawah sudah berhenti. Saat dilakukan pengereman ada perubahan

kecepatan *longitudinal* dan kecepatan putar sebagai efek dari nilai *slip* yang dapat dikendalikan. Ketika terjadi pengereman, kontroler mampu mempertahankan *slip* sebesar 0,22.

BAB 5
PENUTUP

5.1 Kesimpulan

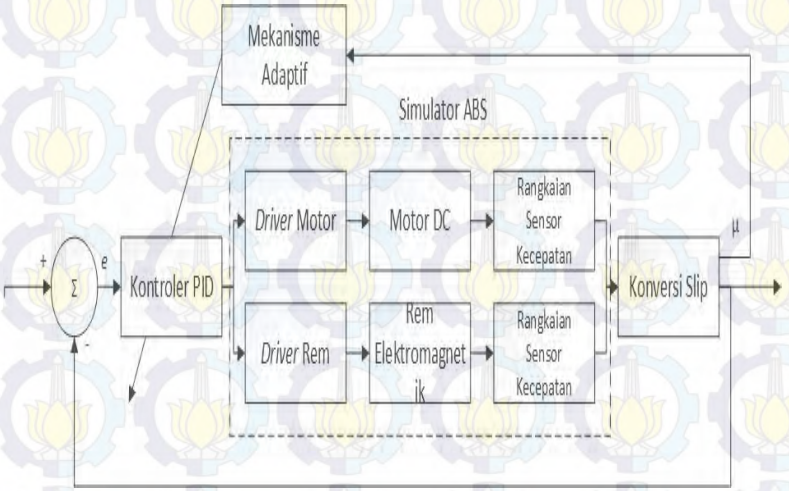
Dari hasil desain kontrol yang telah dikerjakan dan beberapa penerapan dalam tugas akhir ini, dapat diambil kesimpulan sebagai berikut:

a. Perancangan kontroler PID *gain scheduling* dalam simulasi dapat mencapai *set point* yang diinginkan, tidak terdapat *overshoot* dan *error steady state*, serta mempercepat respon *output* orde I disetiap kondisi jalan, jika dibandingkan dengan kontroler PID biasa.
b. Pada saat implementasi kontroler PID *gain scheduling* mampu mempertahankan *slip* yang berada dalam *range slip* 10-30% pada semua kondisi jalan saat terjadi pengereman. Pada $\mu_k \leq 0{,}2$ kontroler PID mampu mempertahankan *slip* dengan nilai 0,21. Pada $0{,}2 < \mu_k \leq 0{,}4$ kontroler PID mampu mempertahankan *slip* dengan nilai 0,14. Pada $\mu_k > 0{,}4$ kontroler PID mampu mempertahankan *slip* dengan nilai 0,22.

5.2 Saran

Untuk kelanjutan riset yang akan datang, diharapkan adanya peralatan untuk menghitung μ, selain itu diharapkan pengembangan dari segi *plant* yang memungkinkan diterapkannya kontroler PID dan kontrol cerdas.

LAMPIRAN A
SKEMA ALAT



--Halaman ini sengaja dikosongkan--

LAMPIRAN B
PROGRAM YANG DIGUNAKAN

a) Program Pwm Motor dan Rem Elektromagnetik

```
int pinrem= 4 ;
int motor = 10 ;
int rem= 9 ;
void setup() {
  // put your setup code here, to run once:
   //pinMode(pinrem,INPUT_PULLUP)
   pinMode(motor,OUTPUT);
  // pinMode(rem,OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:

  //if ( digitalRead(pinrem) == LOW)
   {
     //
      //digitalWrite(rem,HIGH);
      //delayMicroseconds(1998);
      //digitalWrite(rem,LOW);
      //delayMicroseconds(2);

   }
   //else
   {

    digitalWrite(motor,HIGH);
    delayMicroseconds(1400);
    digitalWrite(motor,LOW);
    delayMicroseconds(600);
   }
}
```

b) Program Konversi ke *Slip* dari Sensor kecepatan

```
int encoder1  = 4 ;
int encoder2  = 8 ;

void setup() {
  // put your setup code here, to run once:
   Serial.begin(9600);
   pinMode(encoder1,INPUT);  //longitudinal
   pinMode(encoder2,INPUT);  //putar
   pinMode(13, OUTPUT);
}

void loop() {
    digitalWrite(13, digitalRead(encoder1));

  //if  (digitalRead(encoder2) == HIGH) {

    //while (digitalRead(encoder2) == HIGH)
    {
      //do nothing
    }
   //while (digitalRead(encoder1) == LOW)
    {
     //do nothing
    }
  // kec_putar();
  //}
     if (digitalRead(encoder1) == HIGH )
   {
     while (digitalRead(encoder1) == HIGH)
     {
       //do nothing
     }
    while  (digitalRead(encoder2) == LOW)
     {
       // do nothing
     }
     kec_longitudinal();
```

```
  }
}

void kec_longitudinal()
{
  Serial.print("kecepatan longitudinal =");
  unsigned long waktu = 0;
  float jeda = 0;
  float kecepatan_longitudinal = 0;
  word vel = 0;
  waktu = pulseIn(encoder1, LOW);
  jeda = waktu;
  jeda = jeda / 1000000;
  kecepatan_longitudinal = (1 / (jeda * 110 ))* 60 ;
  vel =  kecepatan_longitudinal;
  Serial.println(vel);
}

void kec_putar()
{
  Serial.print("kecepatan putar =");
  unsigned long waktu = 0;
  float jeda = 0;
  float kecepatan_putar = 0;
  word rpm = 0;
  waktu = pulseIn(encoder2, LOW);
  jeda = waktu;
  jeda = jeda / 1000000;
  kecepatan_putar = (1 / (jeda * 110)) * 60 ;
  rpm =  kecepatan_putar;
  Serial.println(rpm);
}
```

c) Program Arduino Tahap Online

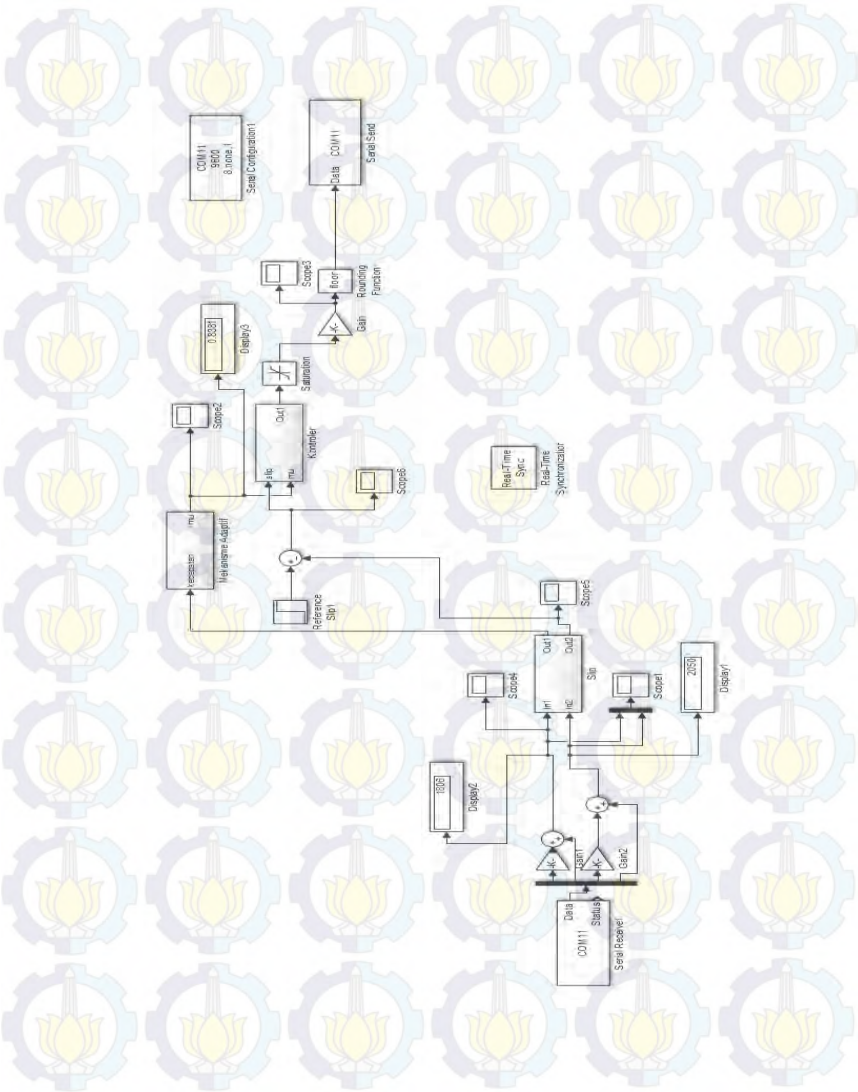
```
#include <Wire.h>
int encoder1 = 7;
```

```
int encoder2= 5;
byte percent ;
float b;
int sinyalkontrol;
char tampilan[1];
char tampilan1[1];
char tampilan2[1];

void setup()
{
  //Wire.begin();
  Serial.begin(9600);
  pinMode(encoder1,INPUT);
  pinMode(encoder2,INPUT);
}

void loop()
{
  //sinyalkontrol=map(percent,0,1,0,255);
  //analogWrite(13,sinyalkontrol);
  if (Serial.available()>0) {
  percent=Serial.read();
  //sinyalkontrol=map(percent,0,1,0,255);
  //Serial.print(sinyalkontrol);
  b = percent;
  analogWrite(9,b);

  bacaEncoderLong();
  bacaEncoderAng();
  Serial.read();}


}

void bacaEncoderLong()
{
```

```
  //Serial.print("kecepatan longitudinal =");
  unsigned long waktu = 0;
  float jeda = 0;
  float kecepatan_longitudinal = 0;
  int vel = 0;
  byte i;
  byte j;
  waktu = pulseIn(encoder1, LOW);
  jeda = waktu;
  jeda = jeda / 1000000;
  kecepatan_longitudinal = (1 / (jeda * 110 ))* 60 ;
  if(kecepatan_longitudinal>2200 && kecepatan_longitudinal==0){
    vel=vel;}
    else{
  vel =  int(kecepatan_longitudinal)+200;}

  i=vel/256;
  j=vel%256;
  Serial.write(i);
  Serial.write(j);
}
void bacaEncoderAng()
{
  //Serial.print("kecepatan putar =");
  unsigned long waktu = 0;
  float jeda = 0;
  float kecepatan_putar = 0;
  int rpm = 0;
  byte i;
  byte j;
  waktu = pulseIn(encoder2, LOW);
  jeda = waktu;
  jeda = jeda / 1000000;
  kecepatan_putar = (1 / (jeda * 110)) * 60 ;
  if(kecepatan_putar>2200 && kecepatan_putar==0){
    rpm=rpm;}
```

```
  else{
 rpm =  int(kecepatan_putar);}
 i=rpm/256;
 j=rpm%256;
 Serial.write(i);
 Serial.write(j);
}
```

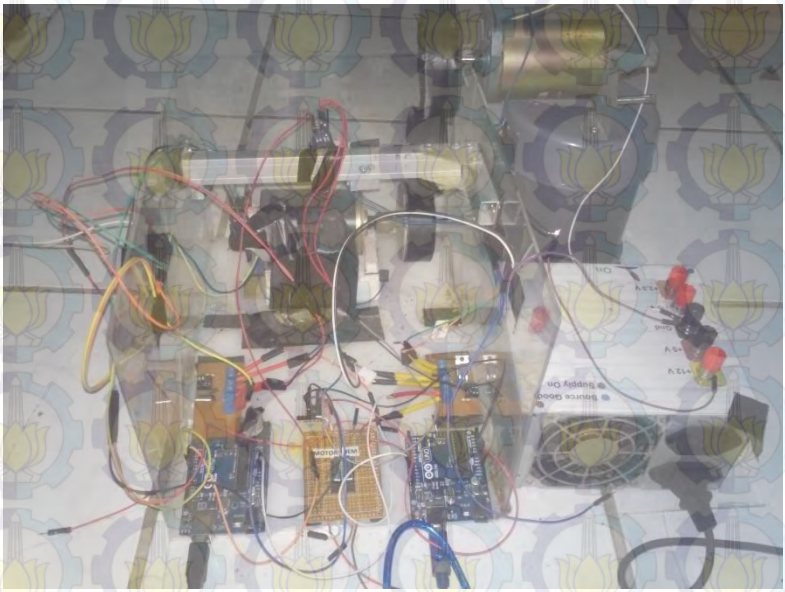
d) Program Blok Simulink Tahap Implementasi



--Halaman ini sengaja dikosongkan--

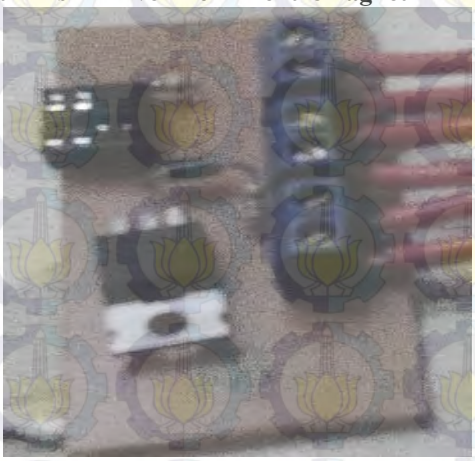
LAMPIRAN C
FOTO ALAT

a. Simulator ABS



b. Bentuk Fisik Driver Motor DC

c. Bentuk Fisik Driver Rem Elektromagnetik



DAFTAR PUSTAKA

[1] H. Mudia, Perancangan dan Implementasi Kontroler PID Adaptif pada Pengaturan Kecepatan Motor Induksi Tiga Fasa, *Tugas Akhir*, Jurusan Teknik Elektro ITS, 2012.

[2] Idar Peterson, Tor A johansen, "Wheel Slip Control in ABS Brakes using Gain Scheduled Constrained LQR," dalam *European Control Conference (ECC)*, Porto, Portugal, 2001.

[3] N.Raesian, N.Khajehpour, and M.Yaghoobi, "A New Approach in Anti-lock Braking System (ABS) Based on Adaptive Neuro-Fuzzy Self-tuning PID Controller," dalam *2nd International Conference on Control, Instrumentation and Automation (ICCIA)*, Moscow, 2011.

[4] ....., "*Sistem Pengereman (Brake System)*," <URL: http://artikel-teknologi.com/sistem-pengereman-break-system>, 2011.

[5] ....., *Rotary Encoder Catalog: North American Edition*, Ohio:Pepperl+Fuchs, Inc, 2007.

[6] D. W, *Arduino Internals*, New York: Appress, 2011.

[7] G. D. Nusantoro, "Identifikasi Sistem Plant Suhu dengan Metode Recursive Least Square," *EECCIS,* vol. 6, no. 1, pp. 67-74, 2012.

[8] M. Ali, "Pembelajaran Perancangan Kontroler PID dengan Software MATLAB," *Jurnal Edukasi,* vol. 1, no. 1, pp. 1-8, 2004.

[9] K. Ogata, *Modern Control Engineering*, Prentice Hall, New Jersey, 2002.

[10] Karl J. Astrom and Bjorn Wittermark, *Adaptive Control*, Mineola: Manufactured in United States of America Dover, 2008.

[11] ....., "*Nisca DC Motor*," <URL: http://www.nisca.co.jp/english/e-mos03.html.> .

[12] SINFONIA, "*SELC&B Dry, Single-Plate Clutches and Brakes*," <URL:http://www.sinfo-t.jp/eng/download/clutch.htm>.

[13] M. Smeja, "*Modelling and Identificaion of Antilock Braking System (ABS)*," AGH University, Slovakia, 2010.

[14] Zuhal, *Dasar Tenaga Listrik*, Bandung: Penerbit Bandung, 1991.

[15] Z. Vukic, "*A Tutorial on Adaptive Control: The Self-Tuning Approach*," University of Zagreb, Croatia, 2000.

[16] K. Kogut, "Anti-lock Braking System Modelling and Parameters Identification," dalam *IEEE Control System*, Miedzyzdroje, 2014.

[17] K. Reif, "Brakes, Brake Control and Driver Assistance Systems Function," *Mechanical Engineering Springer,* vol. VIII, pp. 275-306, 2014.

[18] C. S. a. T. Singh, "Optimal Fuzzy Logic Control For an Anti-Lock Braking System," dalam *IEEE Control System*, New York, 1996.

RIWAYAT HIDUP

Muhammad Fadli Ilmi lahir bulan November di Bandung Jawa Barat, merupakan putra pertama dari 3 bersaudara dari pasangan Wawan Hermawan dan Cicah Rodisah. Menamatkan pendidikan dasar di SDN Karang Pawulang IV Bandung dan melanjutkan ke SMPN 13 Bandung selama 1 tahun, kemudian melanjutkan serta menamatkan di SMPN 1 Balikpapan, Kalimantan Timur. Kemudian menamatkan di SMAN 2 Balikpapan hingga lulus pada tahun 2011. Penulis mengikuti jalur PKM dan diterima di Jurusan Teknik Elektro, FTI-ITS. Selama menjadi mahasiswa, penulis pernah menjadi kepala departemen kaderisasi di organisasi KALAM pada tahun 2013-2014, dan kepala departemen Hubungan Masyarakat di UKM Robotika pada tahun 2013-2014. Pada Januari tahun 2016 penulis mengikuti seminar dan ujian lisan Tugas Akhir di bidang Teknik Sistem Pengaturan, Jurusan Tekni Elektro, FTI-ITS.

Email : muhammad.fadli.ilmi11@gmail.com