ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - TE 141599

PERANCANGAN KONTROLER JARINGAN SARAF TIRUAN BERBASIS *FEEDFORWARD* YANG MENGOPTIMALKAN PID UNTUK *ELECTROMAGNETIC ANTI-LOCK BRAKING SYSTEM* (ABS) PADA KENDARAAN LISTRIK

Benny Adijaya Joesoep
NRP. 2211 100 064

Dosen Pembimbing
Ir. Josaphat Pramudijanto, M.Eng.
Andri Ashfahani, S.T., M.T.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2015

FINAL PROJECT – TE 141599

DESIGN ARTIFICIAL NEURAL NETWORK-BASED FEEDFORWARD OPTIMIZE PID CONTROLLER FOR ELECTROMAGNETIC ANTI-LOCK BRAKING SYSTEM (ABS) ON ELECTRIC VEHICLE

Benny Adijaya Joesoep
NRP. 2211 100 064

Supervisor
Ir. Josaphat Pramudijanto, M.Eng.
Andri Ashfahani, S.T., M.T.

DEPARTEMENT OF ELECTRICAL ENGINEERING
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2015

PERANCANGAN KONTROLER JARINGAN SARAF TIRUAN BERBASIS *FEEDFORWARD* YANG MENGOPTIMALKAN PID UNTUK *ELECTROMAGNETIC ANTI-LOCK BRAKING SYSTEM*(ABS) PADA KENDARAAN LISTRIK

TUGAS AKHIR

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

Menyetujui

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Josaphat Pramudijanto, M.Eng.	Andri Ashfahani, S.T., M.T.
NIP. 196210051990031003	NIP. 2200201405003

SURABAYA
JULI, 2015

PERANCANGAN KONTROLER JARINGAN SARAF TIRUAN BERBASIS *FEEDFORWARD* YANG MENGOPTIMALKAN PID UNTUK *ELECTROMAGNETIC ANTI-LOCK BRAKING SYSTEM* (ABS) PADA KENDARAAN LISTRIK

Nama : Benny Adijaya Joesoep
Pembimbing I : Ir. Josaphat Pramudijanto, M.Eng.
Pembimbing II : Andri Ashfahani, S.T., M.T.

ABSTRAK

Anti-lock braking system (ABS) adalah salah satu teknologi dalam sistem keselamatan mobil yang mencegah roda terkunci saat pengemudi melakukan pengereman mendadak atau menekan pedal rem sangat kuat. Roda yang terkunci berarti slip antara kecepatan putar roda dengan kecepatan longitudinal mobil bernilai 1. Pada saat roda terkunci, daya manuver mobil menjadi berkurang sehingga sulit dikemudikan oleh pengemudi. ABS dapat mempertahankan daya manuver mobil pada saat pengereman dengan cara mengontrol slip roda agar tetap berada pada nilai 0,2, karena pada saat itu nilai koefisien gesek jalannya terbesar, dan mobil masih mudah untuk dikemudikan. Pada kendaraan listrik, pengereman dapat dilakukan dengan rem elektromagnetik sehingga memungkinkan menggunakan kontroler PID untuk menggantikan kontroler *bang – bang* yang umum digunakan pada ABS. Sayangnya kontroler PID menghasilkan performa yang kurang bagus ketika diterapkan pada ABS, akibat efek *non-linear* koefisien gesek jalan terhadap slip roda. Untuk meningkatkan performa kontroler PID digunakan *inverse neural model plant* untuk mengoptimalkan masukan sinyal referensi slip. Pada Tugas Akhir ini, metode kontrol jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan kontroler PID terbukti dapat memperbaiki respon *transient plant. Settling time plant* ± 0,5 % menjadi lebih cepat sepuluh kali jika dibandingkan dengan kontroler PID. Pengujian terhadap metode kontrol ini dilakukan pada *software* MATLAB Simulink dengan memperhatikan dinamika dari simulator ABS.

Kata kunci: ABS, *inverse neural model plant,* PID, simulator ABS

(*halaman ini sengaja dikosongkan*)

DESIGN ARTIFICIAL NEURAL NETWORK-BASED FEEDFORWARD WHICH OPTIMIZE PID FOR ELECTROMAGNETIC ANTI-LOCK BRAKING SYSTEM (ABS) ON ELECTRIC VEHICLE

Name **: Benny Adijaya Joesoep**
1st Supervisor **: Ir. Josaphat Pramudijanto, M.Eng.**
2nd Supervisor **: Andri Ashfahani, S.T., M.T.**

ABSTRACT

Anti-lock braking system (ABS) is one of the technology in car safety system to prevent wheels locking when the driver suddenly braking or press the brake pedal strongly. When the wheels locking slip's value between angular speed and longitudinal car's speed will be one, and manuver ability of the car will be decrease, so car difficult to steer. ABS can mantain manuver ability of the car when braking by control the slip's value become 0.2, because at that time road friction coefficient will be greatest and the car still easy to steer. In electric vehicle, we use electromagnetic brake, so we can use PID's controller to replace bang-bang controller which mostly used in commercial ABS. Unfortunately PID controller lead to limitied performance when applied to ABS, due the non-linear effect of road friction coefficient and slip. In order to improve the perfomance of PID controller, i used inverse neural model plant to optimize the reference input slip. In this final project, artificial neural network-based feedforward which optimize PID controller is approved can improve the transient response of the plant. Settling time ± 0.5 % *of the plant can be faster ten times than PID controller. The test of control method which is used in this final project tested in MATLAB Simulink with regard the dynamics of ABS simulator.*

Key words: *ABS, ABS simulator, inverse neural model plant, PID*

(*halaman ini sengaja dikosongkan*)

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yesus Kristus yang selalu memberikan rahmat dan penyertaan-Nya sehingga penulis dapat menyelesaikan penulisan buku Tugas Akhir dengan judul **"PERANCANGAN KONTROLER JARINGAN SARAF TIRUAN BERBASIS *FEEDFORWARD* YANG MENGOPTIMALKAN PID UNTUK *ELECTROMAGNETIC ANTI-LOCK BRAKING SYSTEM (ABS)* PADA KENDARAAN LISTRIK"** dengan baik dan tepat waktu. Tugas Akhir ini disusun untuk memenuhi sebagian persyaratan untuk menyelesaikan pendidikan Strata-1 pada Bidang Studi Teknik Sistem Pengaturan, 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 kerjasama dari berbagai pihak sehinggala kendala-kendala tersebut dapat diatasi. Untuk itu pada kesempatan ini penulis menyampaikan banyak terimakasih dan penghargaan setingi-tingginya kepada :

1. Ibu dan kakak yang selalu memberikan dukungan, semangat, dan doa kepada penulis, serta ayah yang selalu melindungi dari atas.
2. Bapak Josaphat dan Bapak Andri selaku Dosen Pembimbing atas segala bantuan, perhatian, dan arahan selama pengerjaan Tugas Akhir ini.
3. Saudara Sentosa Sondang, Hendra Antomy, dan Tito Luthfan yang telah meluangkan waktunya dalam pembuatan simulator ABS sehingga simulator ABS yang digunakan dapat berjalan dengan baik.
4. Teman-teman seperjuangan Tim ABS, serta kepada semua pihak yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari bahwa pada penyusunan Tugas Akhir ini mungkin masih terdapat kekurangan. Maka dari itu, penulis bersedia menerima kritik dan saran dari para pembaca. Semoga Tugas Akhir ini dapat bermanfaat khususnya untuk penulis, dan umumnya untuk semua pembaca laporan Tugas Ahir ini

Surabaya, 7 Juli 2015
Penulis

(halaman ini sengaja dikosongkan)

DAFTAR ISI

(halaman ini sengaja dikosongkan)

DAFTAR TABEL

(halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

Pada bab ini akan dijelaskan mengenai latar belakang, permasalahan, tujuan penelitian, sistematika laporan, dan relevansi dari Tugas Akhir yang dilakukan.

1.1 Latar Belakang

Dewasa ini, banyak industri otomotif di dalam maupun luar negeri yang sedang berlomba dalam menciptakan mobil listrik. Industri otomotif mulai beralih menciptakan mobil listrik dikarenakan dua alasan penting yaitu cadangan minyak dunia akan habis 18 tahun lebih awal jika memperhitungkan faktor pertumbuhan penduduk dan peningkatan konsumsi per kapita[1], dan mobil listrik tidak mengeluarkan emisi gas buang berupa gas CO2 yang merupakan penyebab efek *global warming* di bumi. Teknologi yang sudah ada pada mobil bahan bakar juga dapat diterapkan dengan mudah pada mobil listrik juga menjadi alasan tambahan mengapa banyak industri otomotif yang mulai menciptakan mobil listrik. Salah satu teknologi yang dapat diterapkan adalah teknologi *Anti-lock Braking System* (ABS). Teknologi ABS merupakan fitur keselamatan standar yang harus ada pada setiap kendaraan roda empat karena teknologi tersebut dapat membantu pengemudi agar roda tidak terkunci pada saat melakukan pengereman penuh. Dengan adanya teknologi ABS, pengemudi dapat merasa aman dan nyaman ketika melakukan pengereman di berbagai kondisi jalan.

Teknologi ABS sangat diperlukan pada saat pengereman karena pada saat roda terkunci maka daya manuver mobil akan berkurang atau mobil dalam keadaan setengah mengerem. Apabila bagian roda depan yang terkunci maka pengemudi akan merasa berat dalam mengendalikan kemudi dan laju kendaraan akan terganggu sehingga memungkinkan terjadinya *understeer*. Apabila bagian roda belakang terkunci pada saat melakukan pengereman di kecepatan tinggi maka mobil akan kehilangan keseimbangannya sehingga memungkinakan terjadi *oversteer* dan mobil bisa keluar dari jalan serta berputar[2].

Strategi metode kontrol ABS yang saat ini dikembangkan sudah mulai menggunakan *brake-by-wire* dan berganti dari metode kontrol kecepatan menjadi metode kontrol slip. Hal ini tentu saja, memungkinkan banyak metode kontrol yang dapat diterapkan untuk menggantikan kontroler *bang – bang* yang sering digunakan pada ABS[3]. Salah satu

metode kontrol tersebut adalah kontroler PID (*Propotional Integral Derivative*) dan beberapa variasi modenya yang sangat terkenal dan banyak digunakan untuk mengontrol beberapa proses di industri.

1.2 Perumusan Masalah

Proses pengereman dikatakan baik jika jarak mobil untuk berhenti pendek dan mobil masih dapat dikendalikan arahnya oleh pengemudi. Proses pengereman yang baik ini dapat dilakukan dengan mengatur besar slip yang dihasilkan oleh kecepatan putar roda mobil dengan kecepatan *longitudinal* mobil pada nilai 0,2. Nilai 0,2 dipilih bedasarkan kurva koefisien gesek jalan terhadap slip roda, karena pada saat itu koefisien gesek jalan menjadi maksimal dan daya manuver mobil masih besar.

Pada metode kontrol slip dan penggunaan *brake-by-wire*, kontroler PID yang digunakan ternyata menghasilkan performa yang kurang bagus pada saat diterapkan di ABS. Hal ini disebabkan kontroler PID tidak dapat mengatasi efek non linear pada ABS yang dihasilkan dari rem elektromagnetik dan kurva koefisien gesek jalan terhadap slip roda. Maka dari itu, pada penelitian Tugas Akhir ini digunakan metode kontrol jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID untuk memperbaiki respon yang dihasilkan kontroler PID.

1.3 Tujuan

Tujuan dari Tugas Akhir ini akan adalah merancang kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID sehingga dapat meregulasi slip pada nilai 0,2 saat pengereman. Metode kontrol ini terdiri dari dua buah kontroler yaitu kontroler PID dan kontroler jaringan saraf tiruan yang bekerja secara paralel. Kontroler jaringan saraf tiruan diharapkan dapat memperbaiki respon *transient* kontroler PID yang kurang bagus, sementara kontroler PID bekerja untuk menstabilkan *plant* dan menekan gangguan yang terjadi, sehingga proses pengereman dapat berjalan dengan baik.

1.4 Batasan Masalah

Untuk memfokuskan arah penelitian Tugas Akhir ini, maka diambil beberapa batasan masalah yang diantaranya adalah sebagai berikut:

1. *Plant* yang digunakan untuk melakukan pengujian dan pengambilan data adalah simulator ABS skala laboratorium.
2. Roda bagian bawah pada simulator ABS dianggap sebagai jalan aspal dengan kondisi kering.

3. Proses pengereman dilakukan pada saat kecepatan putar mobil telah mencapai kecepatan 2193 *rotation per minute* (rpm).
4. Proses pembelajaran jaringan saraf tiruan dilakukan secara *offline* dengan fungsi aktivasi *sigmoid unipolar.*
5. Proses tunning kontroler PID dilakukan secara *offline* dengan menggunakan metode eksperimental.

1.5 Sistematika Penulisan

Pada Tugas Akhir ini sistematika penulisan dibagi menjadi lima bab, yaitu:

Bab 1 : Pendahuluan

Bab ini meliputi latar belakang, permasalahan, tujuan penelitian, sistematika laporan, dan relevansi.

Bab 2 : Teori Penunjang

Bab ini membahas tinjauan pustaka yang membantu peneliti dalam melakukan penelitian, diantaranya adalah konsep sistem ABS, gaya yang terjadi pada simulator ABS, teori tentang rem mikro elektromagnetik, teori identifikasi *plant* dan teori metode kontrol jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID.

Bab 3 : Perancangan Sistem

Pada bab ini dijelaskan mengenai perancangan simulator sistem ABS yang menggunakan rem mikro elektromagnetik dan sensor *rotary encoder* serta perancangan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID berdasarkan teori pada Bab 2.

Bab 4 : Pengujian dan Analisis Sistem

Bab ini memuat hasil pengujian kontroler berupa simulasi dan implementasi pada sistem dan analisisnya.

Bab 5 : Penutup

Bab ini berisi kesimpulan dan saran dari simulasi yang telah dilakukan.

1.6 Relevansi

Hasil yang diperoleh dari pelaksanaan Tugas Akhir ini diharapkan dapat menjadi referensi untuk perancangan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID untuk *Anti-lock Braking System* (ABS) maupun untuk *plant non-liniear* lainnya.

(halaman ini sengaja dikosongkan)

BAB 2
TEORI PENUNJANG

Pada bab ini akan dijelaskan mengenai teori – teori penunjang yang berhubungan dengan *Anti-lock Braking System* (ABS), simulator ABS, identifikasi sistem, dan perancangan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID.

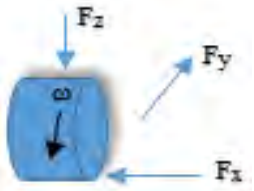
2.1 *Anti-lock Braking System*

ABS adalah salah satu teknologi keselamatan mobil yang mencegah roda berhenti berputar dengan cepat (terkunci) saat pengemudi melakukan pengereman mendadak atau menekan pedal rem sangat kuat. Pada saat roda terkunci atau slip bernilai 1, daya manuver mobil akan berkurang. Persamaan matematika slip roda terhadap jalan secara umum adalah[4]:

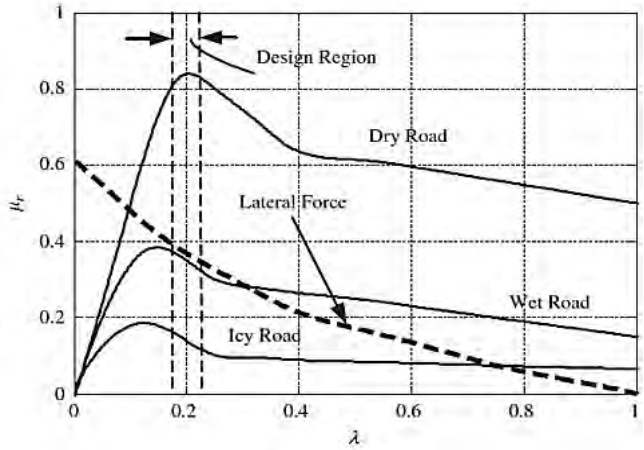
$$\lambda = \frac{v_l - \omega R}{v_l} \tag{2.1}$$

dimana ωR adalah kecepatan putar roda dan v_l adalah kecepatan *longitudinal* mobil. Apabila roda depan yang terkunci maka pengemudi akan merasa berat dalam mengendalikan arah kendaraan sehingga memungkinkan terjadinya *understeer*, tetapi apabila roda belakang yang terkunci maka mobil akan kehilangan keseimbangannya, berputar, dan bisa keluar dari jalan (*oversteer*).[2]

Secara umum, gaya yang bekerja pada ban saat mobil bergerak adalah gaya normal pada ban (F_z), gaya longitudinal (F_y), dan gaya lateral (F_x) seperti yang terlihat pada Gambar 2.1. Gaya normal pada ban akan mempengaruhi gaya *longitudinal* yang disebut koefisien gesek jalan (μ_n). Gaya *longitudinal* ban mempengaruhi jarak mobil berhenti sehingga gesekan maksimum antara ban terhadap jalan dapat diperoleh pada saat gaya *longitudinal* ban maksimum. Sedangkan gaya *lateral* mempengaruhi daya manuver dan kestabilan mobil untuk dikemudikan.



Gambar 2.1 Gaya yang Bekerja pada Ban Saat Mobil Bergerak[4]

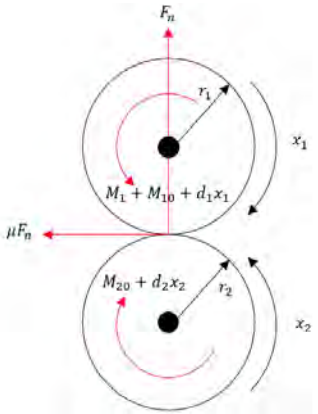


Gambar 2.2 Koefisien Gesek Jalan Terhadap Slip Kendaraan di Berbagai Kondisi Jalan[5]

Maka dari itu, ABS harus dapat mempertahankan gaya *longitudinal* dan *lateral* ban pada daerah yang tepat untuk membuat jarak pengereman seminimal mungkin, dan tetap menjaga daya manuver mobil. Pada Gambar 2.2 dapat diketahui bahwa koefisien gesek maksimum pada berbagai kondisi jalan terjadi pada $\lambda = 0{,}13 - 0{,}2$, dan gaya lateral menurun seiring dengan meningkatkan slip roda terhadap jalan. Maka dari itu, ABS harus dapat meregulasi slip pada daerah dimana koefisien gesek roda terhadap jalan maksimum yaitu 0,18 – 0,22. Nilai 0,18 – 0,22 juga dipilih karena pada saat itu gaya *lateral* pada ban masih dapat membuat mobil mudah untuk dikendalikan arahnya. Untuk melakukan pengujian dan pengoptimalan kinerja metode kontrol slip, maka digunakan simulator ABS skala laboratorium.

2.2 Simulator *Anti-lock Braking System*

Diagram dari Simulator A*nti-lock Braking System* skala laboratorium yang digunakan pada penelitian Tugas Akhir ini dapat dilihat pada Gambar 2.3. Simulator tersebut merepresentasikan model seperempat mobil yang hanya memperhatikan gerakan *longitudinal* dari mobil dan gerakan *angular* dari ban selama proses pengereman. Pada simulator ini kita asumsikan bahwa jari – jari roda selalu bernilai konstan, dan gaya *lateral* yang bekerja pada ban dapat diabaikan.



Gambar 2.3 Sketsa Simulator *Anti-lock Braking System*[6]

Simulator yang digunakan terdiri dari dua buah roda yang berputar dan saling bersinggungan, dimana roda atas dianggap sebagai gerakan roda mobil, dan roda bawah dianggap sebagai gerakan jalan relatif (kecepatan *longitudinal* mobil). Pada simulator ini, roda atas dianggap sebagai roda bebas, sehingga tidak diberi pemberat. Rem elektromagnetik akan dipasang pada roda atas, sedangkan motor *Direct Current* (DC) dihubungkan dengan roda bawah. Kedua komponen elektronik yang terhubung pada roda atas dan roda bawah dikontrol dengan sinyal *Pulse Width Modulation* (PWM). Roda bawah yang dihubungkan dengan motor DC akan mempercepat roda atas untuk mencapai kecepatan putar awal yang diinginkan sebelum melakukan pengereman. 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[7]. Pada simulator ini, kelembaman dari mobil direpresentasikan oleh kelembaman roda bawah, sehingga roda bawah harus memiliki massa yang lebih besar daripada roda atas agar kelembamannya besar. 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 dan penurunan kecepatan pada kedua roda. Perbedaan kecepatan roda atas dan roda bawah ini menghasilkan *slip* yang akan diatur pada penelitian ini.

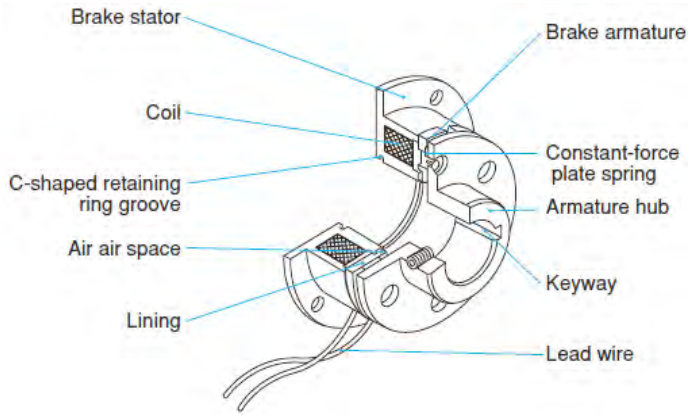
2.3 Rem Mikro Elektromagnetik

Rem mikro elektromagnetik adalah komponen elektronik yang menghasilkan gaya elektromagnetik dan digunakan untuk memperlambat gerakan poros dari suatu benda. Rem elektromagnetik terdiri dari dua bagian yaitu bagian *stator* yang berisi lilitan konduktor dan bagian *armature* yang merupakan piringan dengan bahan logam *non-ferromagnetik* yang ikut berputar seporos dengan benda yang akan diperlambat gerakannya. Konstruksi dari rem mikro elektromagnetik dapat dilihat pada Gambar 2.4.

Prinsip kerja dari rem mikro elektromagnetik adalah ketika arus listrik dialirkan pada sisi *stator*, maka akan timbul medan magnet pada lilitan konduktor. Lalu piringan konduktor pada sisi *armature* yang berputar memotong medan magnet yang timbul pada sisi *stator*. Hasil perpotongan konduktor terhadap medan magnet ini menimbulkan arus *eddy* pada piringan di sisi *armature*. Arus eddy yang timbul akan menghasilkan medan magnet pada sisi *armature* yang arahnya berlawanan dengan medan magnet pada sisi *stator*. Resultan dari kedua medan magnet ini menghasilkan gaya tarik terhadap piringan konduktor dan torsi pengereman, sehingga gerakan piringan konduktor menjadi menurun dan lama kelamaan akan berhenti berputar.[8]

Hubungan antara torsi pengereman yang ditimbulkan oleh rem mikro elektromagnetik dapat ditulis dalam persamaan matematika berikut[9]:

$$T_b = \mu_k \text{ x } r \text{ x } P \qquad (2.2)$$

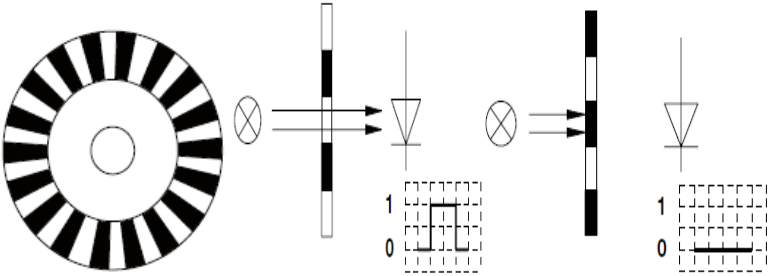


Gambar 2.4 Konstruksi dari Rem Mikro Elektromagnetik[9]

Dari Persamaan (2.2), dapat dilihat bahwa torsi pengereman yang timbul diakibatkan koefisien gesek piringan dan sisi *stator* (μ_k) dikali dengan rata – rata jari – jari benda yang bergesekan (r), dan dikali dengan gaya tarik magnet sisi *stator* terhadap piringan konduktor (P).

2.4 *Rotary Encoder*

Pada penelitian Tugas Akhir ini, peneliti menggunakan dua buah *rotary encoder* untuk mengukur kecepatan putar dan kecepatan *longitudinal* pada simulator ABS. *Rotary encoder* sendiri adalah peralatan yang mengubah besaran rotasi mekanik suatu benda menjadi sinyal listrik agar rotasi tersebut dapat dihitung oleh *counter, tachometer*, dan peralatan kontrol industri semacam *Programmable Logic Control (PLC). 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, bisa juga *rotary encoder* menggunakan piringan dengan beberapa sisi berwarna hitam dan sisi lainnya transparan. 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* terkadang bisa mendeteksi dan tidak bisa mendeteksi cahaya dari LED. Ketika sumber cahaya dapat melewati piringan dan dideteksi oleh *phototransistor, phototransistor* akan mengeluarkan pulsa listrik 5 Volt. Lalu ketika piringan menutup jalur cahaya dari sumber cahaya menuju *phototransistor*, maka *phototransistor* akan mengeluarkan pulsa listrik 0,5 Volt. Cara kerja dari *rotary encoder* dapat dilihat pada Gambar 2.5.



Gambar 2.5. Cara Kerja *Rotary Encoder*[10]

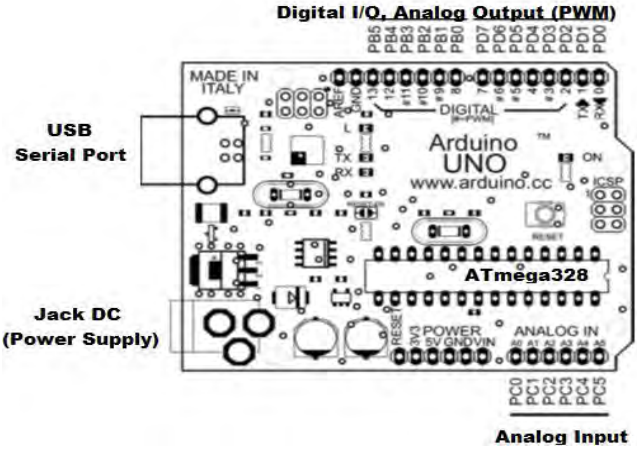
Rotary encoder dapat dibedakan menjadi dua jenis berdasarkan *output* yang dihasilkan, yaitu *incremental rotary encoder*, dan *absolute rotary encoder*. *Incremental rotary encoder* akan menghasilkan sejumlah pulsa listrik pada saat piringan berotasi, sedangkan *absolute rotary encoder* akan menghasilkan kode digital yang menyatakan posisi tertentu dari piringan yang berotasi. Pada *incremental rotary encoder* untuk memperoleh kecepatan putarnya didapatkan dari hubungan frekuensi, *rotation per minute* (rpm), dan banyaknya pulsa tiap revolusi. Hubungan tersebut dapat ditulis dalam persamaan matematika berikut[10]:

$$f = \frac{n}{60} \; x \; Z \tag{2.3}$$

dimana f adalah frekuensi dengan satuan Hertz, n adalah kecepatan putar yang terbaca oleh *rotary encoder* dalam satuan rpm, dan Z adalah banyaknya pulsa yang dihasilkan oleh *rotary encoder* tiap revolusi.

2.5 Arduino Uno

Aduino adalah papan rangkaian elektronik yang bersifat *open source* dan menggunakan *Integrated Circuit* (IC) ATmega328 sebagai mikrokontroler. Arduino Uno pertama kali diperkenalkan pada 25 September 2011 di New York *Maker Faire*[11]. Arduino memiliki bahasa pemrograman sendiri dan memiliki kapasitas memori program sebesar 32 Kbytes (0,5 KBytes memori digunakan untuk *bootloader*). Diagram rangkaian elektronik Arduino dapat dilihat pada Gambar 2.6



Gambar 2.6 Sketsa Rangkaian Elektronik Arduino[11]

Arduino memiliki *Universal Serial Bus* (USB) *serial port* yang menggantikan 9 pin RS-232 serial konektor melalui USB *interface chips*. Arduino menyediakan 2 jenis masukan *power supply*, yaitu melalui USB *serial port* dan melalui *jack* DC *voltage* dengan tegangan *input* yang disarankan 7 – 12 Volt, dan batas tegangan masukannya 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 untuk pin *input/output* (14 pin untuk *digital input/output* dan *analog input* berupa PWM, serta 6 pin untuk *analog input*), 7 pin untuk suplai tegangan (5 Volt, 3,3 Volt, dan *Ground*), dan 1 buah untuk pin *reset*. Arus DC yang dapat dikeluarkan oleh Arduino adalah 40 mA pada tegangan 5 Volt, dan 50 mA pada tegangan 3,3 Volt.

2.6 Identifikasi Sistem

Sebelum kita dapat melakukan pengaturan pada suatu *plant*, terlebih dahulu kita harus mengetahui dinamika dari *plant* tersebut. Dinamika dari suatu *plant* dapat direpresentasikan dalam suatu persamaan matematika yang sering disebut dengan model matematika sistem. Model matematika dari sebuah sistem dapat diperoleh melalui dua cara yaitu permodelan fisik dan identifikasi sistem. Permodelan fisik merupakan pendekatan matematis hukum – hukum *sains* untuk menjelaskan dinamika *plant*. Sedangkan identifikasi sistem adalah usaha untuk mendapatkan model matematika *plant* berdasarkan data masukan dan keluaran *plant*. Proses identifikasi sistem dapat dilakukan secara *online* maupun secara *offline*.

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 (biasanya berupa sinyal *step)*, sedangkan identifikasi dinamis menggunakan sinyal *input* dengan nilai yang berubah terhadap waktu (biasanya berupa sinyal *random)*. Pada Tugas Akhir kali ini, digunakan identifikasi dinamis dengan pemodelan sistem diskrit bentuk deterministik.

2.6.1 Pemodelan Sistem

Pemodelan sistem merupakan langkah yang penting dalam sistem pengaturan, sebab pemodelan sistem yang salah menyebabkan kontroler yang telah didesain dan disimulasikan dalam komputer tidak dapat diimplementasikan pada *plant* yang sesungguhnya. Pemodelan sistem

yang dapat dilakukan pada identifikasi dinamis adalah pemodelan sistem diskrit. Bentuk umum pemodelan sistem diskrit adalah sebagai berikut[12]:

$$A(q)\, y(k) = B(q)\, u(k - n) + C(q)\, \eta(k) \tag{2.4}$$

dimana :

$$A(q) = a_1\, q^{-1} + a_2\, q^{-2} + \ldots + a_{na}\, q^{-na} \tag{2.5}$$

$$B(q) = b_1\, q^{-1} + b_2\, q^{-2} + \ldots + b_{nb}\, q^{-nb} \tag{2.6}$$

$$C(q) = c_1\, q^{-1} + c_2\, q^{-2} + \ldots + c_{nc}\, q^{-nc} \tag{2.7}$$

$$y(k) = \text{keluaran} \tag{2.8}$$

$$u(k) = \text{masukan} \tag{2.9}$$

$$\eta(k) = noise \tag{2.10}$$

Berdasarkan Persamaan (2.4) 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 (2.4) juga dapat dibentuk pemodelan sistem diskrit yang bergantung pada adanya polinomial A, B, dan C yang dapat dilihat pada Tabel 2.1.

Tabel 2.1 Macam – Macam Bentuk Pemodelan Sistem Diskrit

Polinomial yang Diikutsertkan	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)

2.6.2 Metode Estimasi Parameter

Ada beberapa metode identifikasi parameter yang dapat digunakan, yaitu metode identifikasi parameter *liniear* simultan, metode *gradient* dan metode *least square*. Metode *gradient* adalah metode yang diturunkan melalui kriteria *error* estimasi kuadrat minimum yang dapat dilihat pada Persamaan (2.11)[13]. Sedangkan metode *least square* merupakan pengembangan dari metode *gradient* dengan kriteria yang diminimumkan melibatkan semua pengukuran yang dapat dilihat pada Persamaan (2.12) [13].

$$\min J = \min \left\{ \frac{1}{2} [y(k) - \varphi^T (k\text{-}1)\, \theta(k)]^2 \right\} \tag{2.11}$$

$$\min J = \min \left\{ \frac{1}{2} \sum_{i=0}^{k} [y(i) - \varphi^T (i\text{-}1) \ddot{\Theta}(i)]^2 \right\} \tag{2.12}$$

Pada Tugas Akhir ini digunakan metode *least square* dalam estimasi parameter. Metode *least square* memiliki dua algoritma dalam mendapatkan nilai estimasi parameter $\ddot{\Theta}(i)$ 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 $\varphi(i-1)$, $\theta(i-1)$, dan *gain* estimasi. Lalu proses iterasi dilakukan dengan mengukur *input output* dari *plant*, dan dilanjutkan dengan rekonstruksi $\varphi^T(i\text{-}1)$. Selanjutnya menghitung nilai estimasi parameter $\ddot{\Theta}(i)$ 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.

2.6.3 Validasi Model

Setelah pemodelan sistem dengan metode estimasi parameter didapatkan, langkah selanjutnya adalah validasi model yang didapatkan. Validasi model adalah proses membandingkan hasil identifikasi dinamis *plant* dengan hasil pemodelan yang didapatkan dengan metode estimasi parameter tertentu. Salah satu metode validasi model adalah *Root Mean Square Error* (RMSE). Metode RMSE dapat ditulis dalam persamaan matematika sebagai berikut[14]:

$$Root\ Mean\ Square = \sqrt{\frac{1}{n}\sum_{i=1}^{n} y_i - \hat{y}_i} \tag{2.13}$$

dimana n adalah jumlah data, y_i adalah hasil identifikasi dinamis *plant,* dan $\hat{y}_i$ adalah hasil pemodelan.

2.7 Kontroler PID

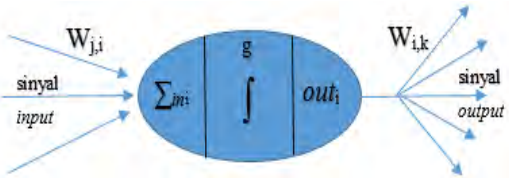
Kontoler *Propotional Integral Derivative* (PID) adalah salah satu kontroler *feedback* yang paling banyak digunakan untuk pengaturan proses di industri[15]. Kontoler PID terdiri dari 3 mode yaitu mode *propotional* (*propotional* terhadap *error*), mode *integral* (*propotional* terhadap *integral error*), dan mode *derivatve* (*propotional* terhadap *derivative error*). Masing – masing mode memiliki konstanta yang memiliki efek tertentu. Mode *propotional* memiliki konstanta Kp yang berfungsi untuk mempercepat respon *transient plant*. Mode *integral* memiliki konstanta Ki yang berfungsi untuk memperbaiki dan menghilangkan *error steady-state*. Mode *derivative* memiliki konstanta Kd yang berfungsi untuk mengurangi *overshoot* dan menstabilkan sistem. Pada Tugas Akhir ini digunakan kontroler mode *propotional-integral-derivative*.

Kontroler mode *propotional-integral-derivative* merupakan kontroler gabungan tiga mode kontroler PID. Kontroler ini akan menghasilkan sinyal kontrol yang lebih baik dari gabungan mode kontroler PID yang lain karena menggunakan tiga mode kontroler PID. Kontroler ini bekerja dengan memperbesar sinyal *error* e(t) dengan konstanta Kp, melakukan *integral* sinyal *error* e(t) dengan konstanta Ki dan melakukan *derivative* sinyal *error* e(t) dengan konstanta Kd. Sinyal kontrol yang dihasilkan kontroler mode *proportional-integral-derivative* (u(t)) dapat dirumuskan dalam persamaan matematika berikut[15]:

$$u(t) = K_p\left(e(t) + \frac{1}{\tau_i}\int_0^t e(t)\,dt + \tau_d \frac{de(t)}{dt} \right) \tag{2.14}$$

2.8 Jaringan Saraf Tiruan

Jaringan saraf tiruan adalah salah satu sistem pemrosesan informasi yang terinspirasi oleh jaringan saraf otak dan diciptakan untuk memecahkan masalah seperti yang terdapat pada pengenalan pola[16].



Gambar 2.7 Cara Kerja *Neuron* Dalam Jaringan Saraf Tiruan[16]

Jaringan saraf tiruan pertama kali diperkenalkan oleh *neurophysiologist* Waren McCulloch dan *logician* Walter Pitts pada tahun 1943. Jaringan saraf tiruan dikembangkan sebagai turunan model matematika dari kesadaran otak manusia, karena jaringan saraf tiruan dapat menirukan cara kerja dan proses pembelajaran otak manusia. Pada jaringan saraf tiruan, *neuron* juga disebut *neuron* atau *node,* serta saling berhubungan dengan yang lainnya dengan bobot teretentu. Maka ketika ada sinyal yang masuk ke dalam *neuron*, sinyal tersebut dikalikan dengan bobot yang bersesuaian serta dijumlahkan. Setelah menjumlahkan sinyal yang masuk, *neuron* akan menghasilkan sinyal *output* melalui suatu fungsi aktivasi. Walaupun masih kurang sempurna, namun cara kerja *neuron* ini mirip dengan cara kerja *neuron* otak yang ada. Cara kerja *neuron* pada jaringan saraf tiruan dapat dilihat pada Gambar 2.7

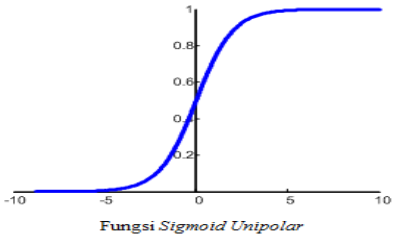
Berdasarkan Gambar 2.7, misalkan ada *n* buah sinyal yang masuk ke *neuron*, maka sinyal keluaran dari *neuron* tersebut dapat dirumuskan dalam persamaan matematika sebagai berikut[16]:

$$in_i = \sum_{n=1}^{n} \text{sinyal } input \text{ x } W_i \qquad (2.15)$$

$$out_i = g(in_i) \qquad (2.16)$$

2.8.1 Fungsi Aktivasi Jaringan Saraf Tiruan

Fungsi aktivasi digunakan untuk mengaktifkan setiap *neuron* agar dapat menghasilkan keluaran terhadap setiap masukan yang ada. Banyak fungsi aktivasi yang dapat digunakan seperti fungsi *hard limit, threshold*, *linear, sigmoid unipolar, dan sigmoid bipolar*. Akan tetapi yang paling banyak digunakan adalah fungsi *sigmoid unipolar* karena bentuk fungsinya dianggap mendekati sinyal kerja otak[16]. Pada Gambar 2.8 dapat dilihat bentuk dari fungsi *sigmoid unipolar.* Persamaan matematika dari fungsi *sigmoid unipolar* dapat direpresentasikan sebagai berikut[16]:



Gambar 2.8 Fungsi Aktivasi *Sigmoid Unipolar*

$$y = \frac{1}{1 + e^{-x}} \tag{2.17}$$

2.8.2 Jaringan *Multi Layer*

Jaringan *multi layer* adalah jaringan saraf tiruan struktur *feedforward* yang mempunyai *neuron* yang tersusun dalam tiga atau lebih lapisan[16]. Lapisan pertama adalah lapisan *input* yang berfungsi untuk menerima dan meneruskan sinyal *input* ke lapisan selanjutnya. Lapisan kedua dan lapisan yang paling banyak adalah lapisan tersembunyi yang menjadi penghubung antara lapisan *input* dan lapisan *output*. Lapisan ketiga adalah lapisan *output* yang akan mengeluarkan tanggapan terhadap semua sinyal *input* yang masuk ke dalam jaringan. Cara kerja dari jaringan saraf *multi layer* adalah sinyal *input* masuk ke dalam jaringan melalui lapisan *input*. Selanjutnya sinyal *input* akan diteruskan menuju lapisan tersembunyi. Pada lapisan tersembunyi, semua sinyal *input* dari lapisan *input* akan dijumlahkan dan dikalikan dengan bobot *neuron*. Sebelum sampai ke lapisan output, hasil penjumlahan sinyal *input* pada lapisan tersembunyi diaktivasi dengan fungsi aktivasi. Selanjutnya sinyal dteruskan dan mencapai lapisan *output*. Pada lapisan *output*, *neuron* akan melakukan proses yang sama pada lapisan tersembunyi sebelum mengeluarkan tanggapan terhadap sinyal *input* yang masuk.

2.8.3 *Backpropagation*

Backpropagation adalah metode *supervised learning* yang biasanya digunakan pada jaringan *multi layer*. Metode *supervised learning* adalah metode pembelajaran yang memerlukan target *output* terhadap suatu *input* untuk dimasukan dalam data proses pembelajaran[16]. *Backpropagation* terdiri dari dua proses yaitu *feedforward* dan *backward*. *Backpropagation* menggunakan *error output* jaringan untuk mengubah

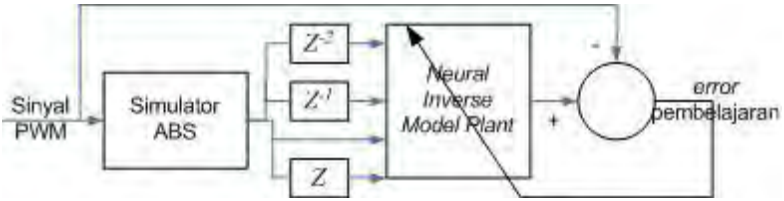
nilai bobot penghubung *neuron*. *Backpropagation* dimulai dengan penginisialisasian bobot awal dengan nilai *random* yang cukup kecil. Selanjutnya metode *backpropagation* melakukan proses *feedforward,* dimana *neuron* pada lapisan *input* menerima sinyal *input* ke-i dan meneruskan sinyal tersebut ke semua *neuron* pada lapisan tersembunyi. *Neuron* pada lapisan tersembunyi akan menjumlahkan sinyal dari lapisan *input* yang telah dikalikan dengan masing – masing bobot penghubung dan ditambahkan dengan bias. Sebelum sampai ke lapisan *output*, hasil penjumlahan sinyal *input* pada lapisan tersembunyi diaktivasi dengan fungsi aktivasi. Selanjutnya sinyal dteruskan dan mencapai lapisan *output*. Pada lapisan *output*, *neuron* akan melakukan proses yang sama pada lapisan tersembunyi sebelum mengeluarkan tanggapan terhadap sinyal *input* yang masuk. Selanjutnya metode melakukan proses *backward* dengan membandingkan *output* jaringan saraf terhadap pola target yang telah ditentukan, jika terdapat perbedaan atau *error* maka terjadi perubahan bobot penghubung pada lapisan tersembunyi dan lapisan *output.* Lalu metode ini akan menghitung *error* pada lapisan tersembunyi dan melakukan koreksi terhadap bobot antara lapisan tersembunyi dan lapisan *input*. Proses ini akan terus berlangsung sampai *output* jaringan saraf tiruan sesuai dengan target pembelajaran yang telah ditentukan atau banyaknya pelatihan telah mencapai batas tertentu**.**

2.8.4 Jaringan Saraf Tiruan Sebagai Kontroler

Secara konsep, jaringan saraf tiruan yang digunakan sebagai kontroler menggunakan *inverse* dari proses *plant* sebagai target pembelajarannya[17]. Hal ini dilakukan agar jaringan saraf tiruan yang dibentuk dapat bertindak sebagai *inverse* dari proses plant atau yang sering disebut sebagai *inverse neural model plant*. Dua macam metode kontrol yang menggunakan jaringan saraf tiruan adalah *direct inverse control* dan *indirect inverse control. Direct inverse control* adalah metode kontrol yang menggunakan *inverse neural model plant* sebagai kontroler, sedangkan *indirect inverse control* adalah metode kontrol yang menggunakan *inverse neural model plant* untuk memprediksi *output* dari *plant*.

Proses untuk membentuk *inverse neural model plant* dimulai dengan mendapatkan persamaan *inverse* proses *plant* yang dapat dilihat dari dinamika *plant*. Secara sederhana, proses yang terjadi pada *plant* dapat dijelaskan dengan persamaan matematika sebagai berikut[17]:

$$y(t+1) = g(y(t), \ldots\ldots, y(t-n+1), u(t), \ldots, u(t-m)) \quad (2.18)$$



Gambar 2.9 *Generalized Architecture*[18]

Berdasarkan Persamaan (2.18), maka dapat diperoleh persamaan *inverse* proses *plant* yang dapat dijelaskan dengan persamaan matematika sebagai berikut[17]:

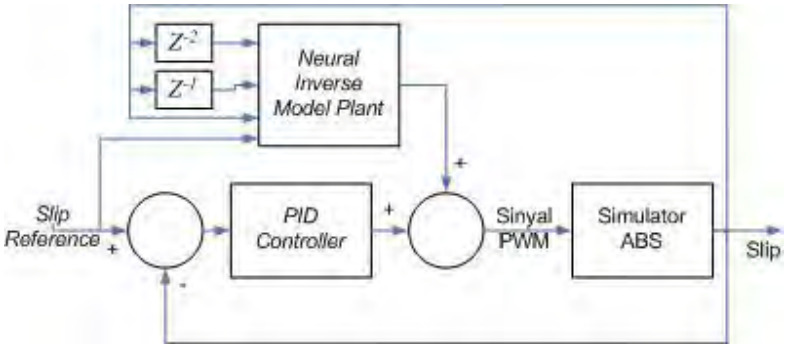
$$u(t) = g^{-1}(y(t+1), y(t), \ldots, y(t-n+1), u(t-1), u(t-m)) \quad (2.19)$$

Setelah mendapatkan persamaan *inverse* proses *plant*, selanjutnya jaringan perlu melakukan proses pembelajaran agar dapat menjadi *inverse neural model plant*. Dua metode untuk melakukan proses pembelajaran terhadap *inverse neural model plant* adalah pembelajaran menggunakan *generalized architecture* dan pembelajaran menggunakan *specialized architecture*. Pelatihan menggunakan *generalized architecture* dapat dilihat pada Gambar 2.9.

Pada pembelajaran menggunakan *generalized architecture* sinyal *u* diberikan kepada *plant*, sehingga *plant* akan menghasilkan *output* berupa sinyal *y* yang juga akan menjadi *input* bagi *inverse neural model plant*. Keluaran dari jaringan akan dibandingkan dengan sinyal u yang diberikan kepada *plant*, jika terdapat *error*, maka sinyal *error* tersebut akan digunakan untuk memperbaiki bobot jaringan. Sedangkan pada pelatihan menggunakan *specialized architecture* sinyal *error* yang dihasilkan dari perbedaan *set point* dan keluaran *plant* digunakan untuk memperbaiki bobot jaringan. Pada pelatihan menggunakan *specialized architecture*, jaringan menggunakan matrik Jacobian untuk memperbaiki bobot yang ada.

2.9 Jaringan Saraf Tiruan Berbasis *Feedforward* yang Mengoptimalkan Kontroler PID

Jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan kontroler PID adalah metode kontrol yang menggabungkan secara paralel jaringan saraf tiruan yang berlaku sebagai kontroler *feedforward* dengan kontroler PID yang berlaku sebagai kontroler *feedback* dalam metode ini.



Gambar 2.17 Struktur Metode Kontrol Jaringan Saraf Tiruan yang Mengoptimalkan Kontroler PID[19]

Pada metode kontrol ini, sinyal kontrol *feedforward* yang akan ditambahkan dengan sinyal kontrol *feedback* sebelum sinyal kontrol sistem masuk ke dalam *plant*. Metode kontrol ini memiliki tujuan untuk memperbaiki respon *transient* yang dihasilkan oleh kontroler *feedback* agar menghasilkan respon *plant* yang lebih cepat. Struktur metode kontrol jaringan saraf tiruan yang mengoptimalkan kontroler yang sudah ada dapat dilihat pada Gambar 2.22. Pada metode kontrol ini, jaringan saraf tiruan yang digunakan merupakan *inverse neural model plant*. Jaringan saraf tiruan yang digunakan pada metode kontrol ini berfungsi untuk memberikan sinyal kontrol *feedforward* langsung dari sinyal *reference* sehingga dapat mempercepat respon *plant* untuk mencapai *setpoint*. Sedangkan kontroler PID digunakan untuk menstabilkan respon *plant* dan meminimalkan gangguan yang ada. Akan tetapi metode kontrol ini juga memiliki kelemahan, yaitu penggunaan *inverse neural model plant* yang tidak tepat akan menyebabkan respon *plant* akan mengalami *overshoot* yang besar dan memerlukan waktu yang lama untuk mejadi stabil.

(*halaman ini sengaja dikosongkan*)

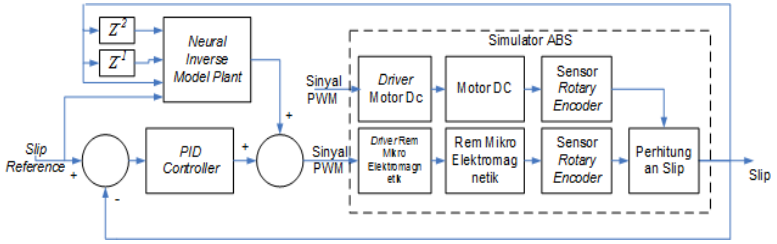
BAB 3
PERANCANGAN SISTEM

Pada bab ini akan dijelaskan perancangan sistem secara keseluruhan. Hal ini termasuk perancangan *hardware*, perancangan *software*, dan perancangan kontroler yang digunakan untuk meregulasi slip simulator ABS (*Anti-lock Braking System*) menjadi 0,2. Perancangan *hardware* yang dilakukan meliputi perancangan mekanik dan elektronik komponen penunjang simulator ABS, sedangkan perancangan *software* yang dilakukan meliputi pembuatan program agar komponen – komponen yang digunakan dapat saling bekerja sama dan dapat melakukan simulasi *anti-lock braking system.*

3.1 Gambaran Umum Sistem

Simulator ABS merupakan peralatan untuk melakukan pengujian dan pengoptimalan kinerja metode kontrol slip. Simulator ABS merepresentasikan model seperempat mobil yang hanya memperhatikan gerakan *longitudinal* dari mobil dan gerakan *angular* dari ban selama proses pengereman. Simulator ABS ini menggunakan dua buah roda yang diletakan secara vertikal dan saling bersinggungan. Roda atas dianggap sebagai gerakan roda mobil, dan roda bawah dianggap sebagai gerakan jalan relatif (kecepatan *longitudinal* mobil). Pada simulator ini, kelembaman dari mobil direpresentasikan oleh kelembaman roda bawah, sehingga roda bawah harus memiliki massa yang lebih besar daripada roda atas agar kelembamannya besar.

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 pada simulator ABS ini dapat dilakukan dengan cara menghilangkan suplai sinyal PWM ke motor DC dan mulai memberikan suplai 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.



Gambar 3.1 Diagram Blok Sistem Pengaturan Slip pada Simulator ABS

Perbedaan masa roda atas dan roda bawah simulator ABS, menyebabkan perbedaan kelembaman dari kedua roda tersebut. Perbedaan kelembaman ini mengakibatkan terjadi perbedaan perlambatan atau penurunan kecepatan pada kedua roda, sehingga menghasilkan *slip* yang dapat dihitung. Selain perbedaan kelembaman antara kedua roda yang mempengaruhi besarnya slip simulator ABS, ada beberapa parameter juga yang mempengaruhi, seperti diantaranya koefisien permukaan roda, dan torsi pengereman yang diberikan pada roda atas.

Pada simulator ini digunakan kontroler jaringan saraf tiruan yang mengoptimalkan kontroler PID. Kontroler ini bertujuan untuk mengatur besar *slip* yang dihasilkan oleh simulator ABS agar selalu bernilai 0,2. Kontroler jaringan saraf tiruan akan memperbaiki respon *transient plant* yang tidak dapat diperbaiki oleh kontroler PID, sedangkan kontroler PID akan menstabilkan respon *plant* dan menekan gangguan yang terjadi. Kontroler PID yang digunakan pada sistem ini merupakan kontroler PID gabungan mode *propotional, integral,* dan *derivative,* sedangkan kontroler jaringan saraf tiruan yang digunakan merupakan *inverse neural model plant*. Blok diagram dari sistem pengaturan slip pada simulator ABS dapat dilihat pada Gambar 3.1

3.2 Perancangan *Hardware*

Pada tahap perancangan *hardware*, terdapat 2 jenis perancangan yang dilakukan, yaitu perancangan mekanik dan perancangan elektronik. Perancangan mekanik merupakan perancangan yang bertujuan menempatkan komponen – komponen yang digunakan dalam simulator ABS secara tepat, sehingga simulator ABS dapat digunakan untuk melakukan pengujian dan pengoptimalan metode kontrol slip. Sedangkan perancangan elektronik merupakan perancangan pembuatan *driver* dan

rangkaian sensor yang membantu agar simulator ABS dapat berjalan dengan sempurna.

3.2.1 Perancangan Mekanik

Perancangan mekanik pada simulator ABS yang digunakan untuk Tugas Akhir ini dibagi menjadi dua bagian yaitu perancangan bagian atas simulator, dan perancangan bagian bawah simulator. Komponen – komponen yang digunakan dalam simulator ABS ini adalah dua buah roda beserta ban mobil *remote control* dengan diameter 3,5 cm, motor DC, rem mikro elektromagnetik, *shaft*, dan dua buah sensor *rotary encoder.*

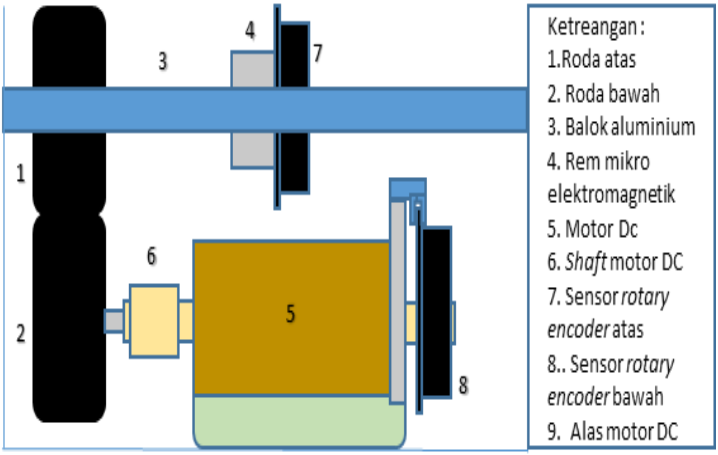
(1) Bagian Atas Simulator ABS

Perancangan mekanik pada bagian atas simulator ABS meliputi penempatan roda beserta ban mobil *remote control*, rem mikro elektromagnetik, dan sensor *rotary encoder* pada satu *shaft*. *Shaft* yang digunakan berbahan dasar *stainless steel* dengan diameter 4 mm dan panjang 22 cm. Roda berserta ban mobil *remote control* yang dipasang pada bagian atas merepresentasikan roda mobil yang berputar. Roda ini juga dianggap sebagai roda bebas sehingga tidak dihubungkan dengan sumber putar (motor DC). Maka dari itu, roda bebas ini harus diletakkan bersinggungan dan tepat di atas roda bagain bawah simulator ABS, agar roda bebas dapat berputar dan dilakukan proses pengereman dengan metode kontrol slip tertentu.

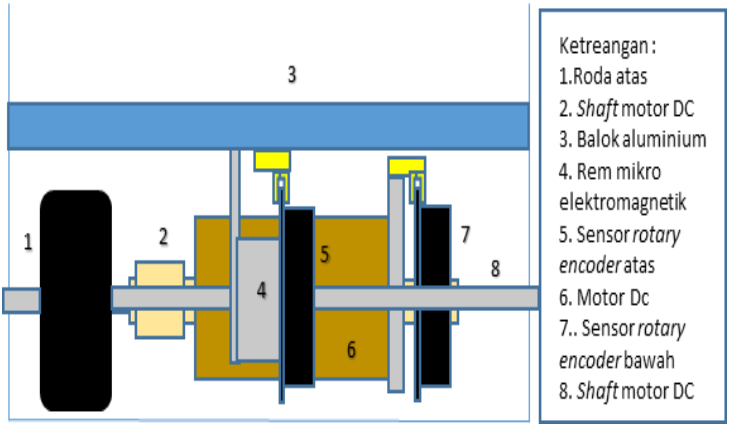
Pada bagian dalam roda roda bebas, dipasang sebuah sensor *rotary encoder* dan rem mikro elektromagnetik. Sensor *rotary encoder* yang digunakan terdiri dari piringan *encoder* dan rangkaian sensor *optocoupler*. Piringan *encoder* yang digunakan berbahan dasar plastik berwarna hitam, berdiameter 6,4 cm, dan terdiri dari 55 lubang. *Rangkaian* sensor *optocoupler* ditempelkan pada balok aluminium yang dipasang sejajar dengan *shaft* roda bebas. Piringan *encoder* diletakan di tengah sensor *optocoupler* serta dikunci pada *shaft* sehingga piringian tidak bergoyang – goyang dan berpindah saat roda berputar. Rem mikro elektromagnetik diletakan di sisi dalam piringan *encoder*, sehingga pada saat rem bekerja timbul arus eddy dan gaya yang berlawanan pada piringan *encoder*. Gaya yang berlawanan dengan arah putar roda bebas menyebabkan perlambatan pada roda bebas. Perlambatan tersebut menyebabkan slip antara roda bebas dan roda bagian bawah simulator. Pengaturan slip antara roda bebas dan roda bagian bawah simulator dilakukan

dengan mengatur besar kecilnya arus yang masuk ke rem mikro elektromagnetik.

Sketsa tampak depan simulator ABS dapat dilihat pada Gambar 3.2, sedangkan sketsa tampak atas simulator ABS dapat dilihat pada Gambar 3.3.



Gambar 3.2 Tampak Depan Sketsa Simulator ABS

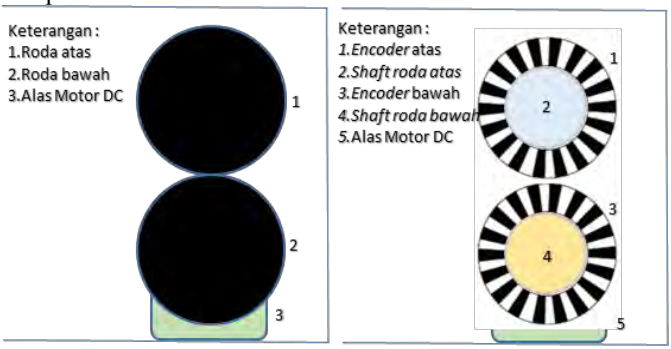


Gambar 3.3 Tampak Atas Sketsa Simulator ABS

(2) Bagian Bawah Simulator ABS

Perancangan mekanik pada bagian bawah simulator ABS meliputi penyatuan roda mobil *remote control* dengan motor DC, dan penempatan sensor *rotary encoder* pada *shaft* motor DC. Roda berserta ban mobil *remote control* yang dipasang pada bagian bawah merepresentasikan kecepatan relatif jalan (kecepatan *longitudinal* mobil). Maka dari itu, roda ini memilik massa yang lebih besar daripada roda pada bagian atas simulator ABS, agar pada saat pengereman memiliki kelembaman yang lebih besar.

Roda bagian bawah simulator ABS disambungkan dengan *shaft* motor DC sehingga dapat menggerakan roda bagian atas sampai mencapai kecepatan tertentu sebelum simulator melakukan proses pengereman. Pada bagian ujung *shaft* motor DC yang tidak bersambungan dengan roda, dipasang sebuah sensor *rotary encoder*. Sensor *rotary encoder* yang digunakan terdiri dari piringan *encoder* dan rangkaian sensor *optocoupler*. Piringan *encoder* yang digunakan berbahan dasar plastik berwarna hitam, berdiameter 6,4 centimeter, dan terdiri dari 55 lubang. Rangkaian sensor *optocoupler* ditempelkan pada rangka motor DC, dan piringan encoder diletakan di tengah sensor *optocoupler* serta dikunci pada *shaft*. Sketsa tampak samping kanan dan kiri simulator ABS dapat dilihat pada Gambar 3.4



Gambar 3.4 Tampak Samping Kiri Dan Kanan Sketsa Simulator ABS

3.2.2 Perancangan Elektronik

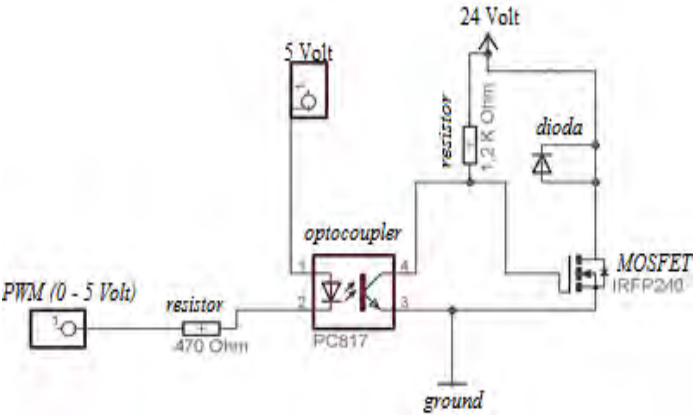
Perancangan elektronik komponen simulator ABS yang digunakan pada Tugas Akhir ini dibagi menjadi tiga bagian yaitu perancangan *driver* motor DC, *driver* rem mikro elektromagnetik, rangkaian sensor *optocoupler*.

(1) *Driver* Motor DC

Rangkaian *driver* motor DC digunakan untuk menjalankan motor DC yang menjadi penggerak utama dalam simulator ABS. Adapun motor DC yang digunakan adalah motor DC buatan Nisca *Corporation* dengan spesifikasi antara lain: tipe motor brush; tegangan pengoperasinya 24 Volt; arus motor 1,123 Ampere; efisiensi motor 71,1%; kecepatan maksimum motor 2500 *rotation per minute* (rpm); dan jenis magnet motor 2 *pole* magnet permanen. *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 keluarannya.

Cara kerja dari rangkaian *driver* motor DC ini adalah Arduino memberikan sinyal PWM ke rangkaian motor DC. Ketika Arduino memberikan pulsa *high* maka *Light Emitting Dioda* (LED) *optocoupler* tidak akan menyala sehingga *transistor optocoupler* menjadi *open* dan tegangan Vcc motor mengalir ke MOSFET. Akibat MOSFET mendapatkan tegangan Vcc motor sebesar 24 Volt 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.

Gambar *skematik* rangkaian *driver* motor DC dapat dilihat pada Gambar 3.5.

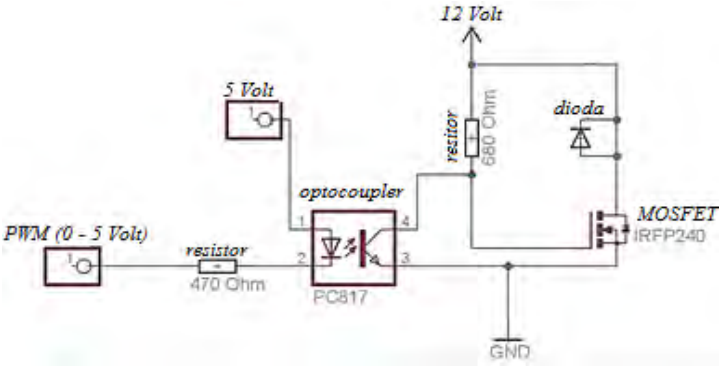


Gambar 3.5 Rangkaian *Driver* Motor DC

(2) *Driver* Rem Mikro Elektromagnetik

Rangkaian *driver* rem mikro elektromagnetik digunakan untuk menjalankan rem yang menjadi aktuator dalam metode kontrol slip di simulator ABS. Adapun rem yang digunakan adalah rem mikro elektromagnetik buatan Shinko Electro dengan spesifikasi antara lain: jenis rem *micro dry-type single-plate*; tegangan pengoperasinya 12 Volt; daya yang dibutuhkan rem sebesar 3,3 Watt; diameter dalam rem 2,7 cm; dan diameter luar rem 1,1 cm. *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 dengan rentang tegangan 0 – 5 Volt yang akan diubah menjadi PWM dengan rentang tegangan 0 – 12 Volt sebagai tegangan keluarannya.

Cara kerja dari rangkaian *driver* rem mikro elektromagnetik ini adalah Arduino memberikan sinyal PWM 0 – 5 Volt ke rangkaian *driver* 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 sebesar 12 Volt akan mengalir ke MOSFET.



Gambar 3.6 Rangkaian *Driver* Rem Mikro Elektromagnetik

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. Pada rangkaian driver rem mikro elektromagnetik ini juga menggunakan dioda sebagai penyearah tegangan bolak balik akibat tegangan yang dihasilkan *optocoupler* sehingga MOSFET tidak terlalu bekerja keras atau cepat panas.

Gambar *skematik* rangkaian *driver rem mikro elektromagnetik* dapat dilihat pada Gambar 3.6.

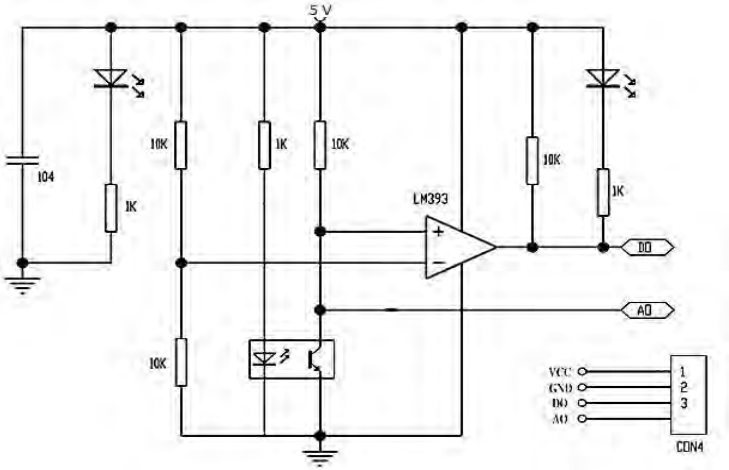
(3) Rangkaian Sensor *Optocoupler*

Sensor *octocoupler* adalah sensor yang mampu mendeteksi kecepatan putar piringan *encoder* berdasarkan ada atau tidaknya cahaya inframerah LED yang diterima oleh *phototransistor.* Rangkaian ini dipasang pada *shaft* motor DC untuk mendeteksi kecepatan putar roda bawah dan pada *shaft* roda bebas untuk mendeteksi kecepatan putar roda atas. Rangkaian sensor *optocoupler* yang digunakan terdiri dari sebuah sensor *optocoupler* dengan lebar celah 5 mm, *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 keluaran yang benar – benar pulsa *digital* dengan arus maksimal 15 mili Ampere.

Cara kerja dari rangkaian sensor *octocoupler* adalah ketika roda bebas dan piringan *encoder* yang terdapat pada satu poros mulai berputar, LED yang mendapatkan tegangan *input* 5 V mulai dapat mengirimkan cahaya inframerah ke *phototransistor.* Terkadang cahaya dari LED tersebut dapat diterima maupun tidak oleh *phototransistror.* 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 rpm.

Gambar *skematik* rangkaian sensor *optocoupler* dapat dilihat pada Gambar 3.7.

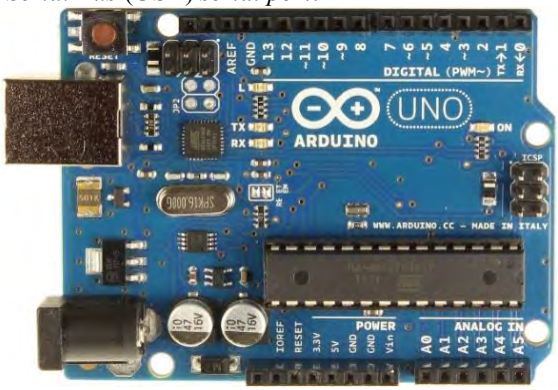


Gambar 3.7 Sensor *Rotary Encoder* yang Terpasang pada *Shaft*

(4) Arduino Uno R3

Arduino adalah papan rangkaian elektronik yang menggunakan mikrokontroler Atmega 328. Simulator ABS yang digunakan pada Tugas Akhir ini, menggunakan Arduino Uno R3 untuk mengolah data hasil sensor *rotary encoder* dan mengeksekusi sinyal kontrol yang telah diolah oleh komputer. Papan rangkaian ini memiliki 14 *input*/*output* pin (6 pin sebagai PWM *output*), 6 *input* analog, 16 MHz *crystal* osilator, dan koneksi kabel USB. Arduino dipilih sebagai penghubung antara komputer dan plant karena pemrograman pada Arduino menggunakan bahasa C yang mudah dipahami oleh semua orang. Bentuk fisik papan rangkaian elektronik Arduino Uno R3 dapat dilihat pada Gambar 3.8

Pada Tugas Akhir ini, Arduino digunakan untuk mengolah data hasil pembacaan sensor *rotary encoder*. Hasil pembacaan dari rotary encoder merepresentasikan kecepatan putar dan kecepatan *longitudinal* simulator ABS. Selain itu, Arduino juga digunakan untuk mengeluarkan sinyal PWM pada *driver* motor DC dan *driver* rem mikro elektromagnetik. Sinyal PWM pada driver motor DC digunakan agar simulator ABS segera mencapai kecepatan awal sebelum melakukan proses pengereman, sedangkan sinyal PWM pada *driver* rem mikro elektromagnetik digunakan sebagai sinyal kontrol kepada simulator. Pengaturan *duty cycle* PWM *driver* rem mikro elektromagnetik disesuaikan dengan sinyal kontrol yang dibutuhkan oleh simulator. Pengiriman data dari papan rangkaian elektronik Arduino ke komputer dan sebaliknya dilakukan melalui *Universal Serial Bus* (USB) *serial port*.



Gambar 3.8 Arduino Uno R3[11]

3.3 Perancangan *Software*

Perancangan *software* merupakan perancangan yang penting setelah mekanik dari suatu *plant* terbentuk, sebab *software* menjadi penghubung antara *plant* dan kontroler (bisa berupa komputer atau mikrokontroler) yang terkadang tidak dapat langsung berinteraksi dengan *plant.* Perancangan *software* dalam Tugas Akhir ini dibagi dalam tiga bagian 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. *Software* MATLAB digunakan untuk keperluan identifikasi sistem dan mengirimkan sinyal kontrol pada saat proses implementasi. Selain itu, MATLAB digunakan juga sebagai *software* untuk desain, simulasi kontroler yang akan digunakan pada pengaturan slip Tugas Akhir ini.

3.3.1 Perancangan Program Pembacaan Sensor *Rotary Encoder*

Perancangan program pembacaan sensor merupakan hal yang paling utama dan penting, sebab jika hasil pembacaan dari sensor tidak tepat maka sulit bagi kontroler untuk memberikan sinyal kontrol 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.

Pada *software Integrated Development Environment* (IDE) Arduino, tidak ada fungsi untuk menghitung banyaknya pulsa listrk yang masuk ke Arduino, melainkan hanya terdapat fungsi untuk menghitung lamanya suatu pulsa yaitu fungsi *pulseIn*. *PulseIn* akan membaca pulsa *high* atau *low* pada suatu pin. Jika misalnya kita akan menghitung lamanya pulsa *high* yang masuk, maka fungsi *pulseIn* akan menunggu pulsa pada pin tertentu menjadi *high*, dan mulai menghitung waktu pulsa tersebut sampai

pulsa menjadi *low* kembali. Akan tetapi, fungsi *pulseIn* ini juga memiliki kekurangan karena tidak dapat menghitung pulsa yang lamanya lebih dari 3 menit, fungsi *pulseIn* hanya dapat membaca pulsa yang lamanya 10 mikro detik sampai 3 menit. Sintaks untuk memanggil fungsi *pulseIn* adalah *pulseIn(pin, value)* atau *pulseIn(pin, value, timeout)*, dimana pin adalah pin berapa pulsa yang masuk akan dihitung lama waktunya, dan *value* adalah jenis pulsa apa yang akan dihitung lama waktunya (*high* atau *low)*. Sedangkan *timeout* adalah waktu untuk menunggu pulsa untuk berubah menjadi *high* atau *low* (biasanya tidak perlu diisi tetapi jika menginginkannya, maksimal dapat diisi satu detik).

Hasil pembacaan *pulseIn* Arduino dalam satuan mikro detik. Selanjutnya hasil pembacaan harus dikali dua, untuk mengetahui lamanya pulsa listrik yang masuk ke dalam pin (dengan asumsi pulsa listrik yang masuk memiliki lebar yang sama antara pulsa *high* dan pulsa *low*) atau periode pulsa listrik yang masuk. Menggunakan rumus fisika dasar, kita dapat memperkirakan banyaknya pulsa yang masuk tiap menit, yaitu dengan menggunakan Persamaan (3.1). Selanjutnya, karena dalam satu putaran tiap menit sensor menghasilkan pulsa listrik sebanyak 55 kali, maka kita dapat mengkonversi banyaknya pulsa yang masuk tiap menit menjadi rpm dengan Persamaan (3.2).

$$\text{banyaknya pulsa tiap menit } (n) = \frac{60}{\text{periode pulsa}} \tag{3.1}$$

$$\text{kecepatan putar (rpm)} = \frac{n}{55} \tag{3.2}$$

Pada Tugas Akhir ini, jenis pulsa lisrtik yang akan dibaca lama waktunya adalah pulsa *low*. Hal ini didasarkan karena pada saat motor DC berputar dengan kecepatan yang sangat cepat, sensor *optocoupler* hanya dapat menghasilkan pulsa listrik dengan rentang tegangan 0 Volt sampai maksimal 3,5 Volt. Sementara itu, fungsi *digitalRead* Arduino tidak dapat membaca sinyal *digital* selain sinyal *digital* dengan tegangan 0 Volt dan 5 Volt. Pada Tugas Akhir ini juga, pin keluaran *analog* sensor *rotary encoder* dihubungkan dengan pin *digital* 3 untuk kecepatan putar roda atas dan pin *digital* 4 untuk kecepatan putar roda bawah. Hal ini dilakukan karena tegangan keluaran dari IC komparator 393 yang terdapat pada sensor kurang akurat. Hasil rancangan *software* pembacaan sensor *rotary encoder* dapat dilihat pada Gambar 3.9 dan lampiran.

Gambar 3.9 Hasil Rancangan Program Pembacaan Sensor *Rotary Encoder*

3.3.2 Perancangan Program 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. Perancangan sinyal PWM sangat berguna sebab hal ini sangat diperlukan dalam proses untuk mendapatkan model sistem melalui identifikasi dinamis. Sinyal PWM adalah sinyal yang terdiri dari dua *state* tegangan (biasanya 0 Volt dan 5 Volt pada Arduino) dengan waktu masing – masing state yang bisa diatur. Misalnya, kita memiliki sinyal PWM 50% dengan lebar keseluruhan pulsa

2 mili detik, maka lebar pulsa *state* 0 Volt 1 mili detik dan lebar pulsa *state* 5 Volt 1 mili detik, jika sinyal PWM 10 % maka lebar pulsa *state* 0 Volt 1,8 mili detik dan lebar pulsa *state* 5 Volt 0,2 mili detik. Sinyal PWM yang dihasilkan oleh Arduino memiliki frekuensi 500 Hz. Pada Tugas Akhir ini, sinyal PWM Arduino akan menjadi *input* bagi rangkaian *driver* motor DC dan rem mikro elektromagnetik untuk diperbesar tegangannya. Sinyal PWM yang dibutuhkan untuk dapat mengaktifkan motor DC memiliki *state* tegangan 0 Volt dan 24 Volt, sedangkan sinyal PWM yang dibutuhkan untuk dapat pmengaktifkan rem mikro elektromagnetik memiliki *state* tegangan 0 Volt dan 12 Volt.

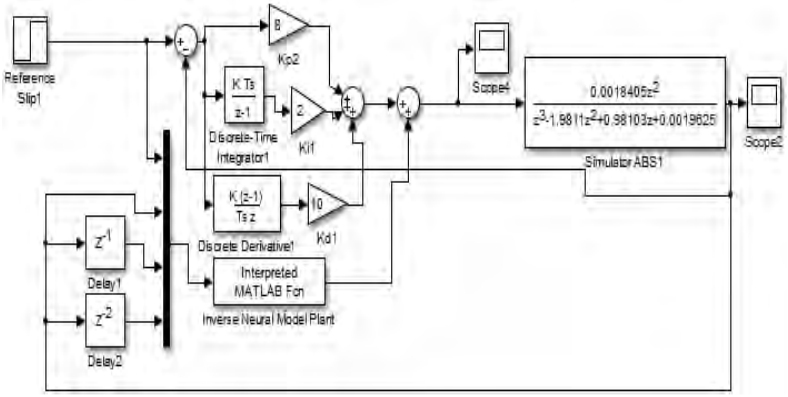
Pada *software* IDE Arduino, sebenarnya terdapat fungsi untuk membuat sinyal PWM yaitu *analogWrite*. Fungsi *analogWrite* memiliki rentang nilai pemanggilan dari 0 sampai 255, dimana jika kita memanggil dengan nilai 255, maka Arduino akan mengeluarkan sinyal PWM 100 %. Sedangkan jika kita memanggil fungsi analogWrite dengan nilai 127, maka Arduino akan mengeluarkan sinyal PWM 50 %. Sintaks untuk memanggil fungsi *analogWrite* adalah *analogWrite(pin, value)*, dimana pin adalah pada pin berapa sinyal PWM akan dikeluarkan, dan *value* adalah rentang nilai pemanggilan fungsi yang menghasilkan sinyal PWM yang berbeda.

Pada Tugas Akhir in, penulis tidak menggunakan fungsi *analogWrite* untuk membangkitkan sinyal PWM, melainkan menggunakan fungsi *digitalWrite* Arduino yang di-*delay* waktu keluarnya dengan lebar pulsa keseluruhan 2 mili detik. Sinyal dengan lebar pulsa keseluruhan 2 mili detik dipilih, karena sinyal PWM yang dihasilkan oleh Arduino memiliki frekuensi 500 Hz atau satu sinyal memililiki periode 0,002 detik. Maka dari itu, hasil keluaran sinyal PWM modifikasi penulis dengan fungsi *digitalWrite* memiliki bentuk dan nilai keluaran yang sama dengan sinyal PWM fungsi *analogWrite*. Pada *software* ini juga dirancang jika pedal rem (berupa *switch on off)* ditekan, maka sinyal PWM motor DC tidak akan dibangkitkan dan sinyal PWM rem mikro elektromagnetik baru akan dibangkitkan, begitu juga sebaliknya jika pedal rem dilepas. Pada Tugas Akhir ini juga, pin *input driver* motor DC dihubungkan dengan pin *digital* 9 dan pin *input driver* rem mikro elektromagnetik dihubungkan dengan pin *digital* 10. Hasil rancangan *software* sinyal PWM untuk motor DC dan rem mikro elektromagnetik dapat dilihat pada Gambar 3.10 dan lampiran.

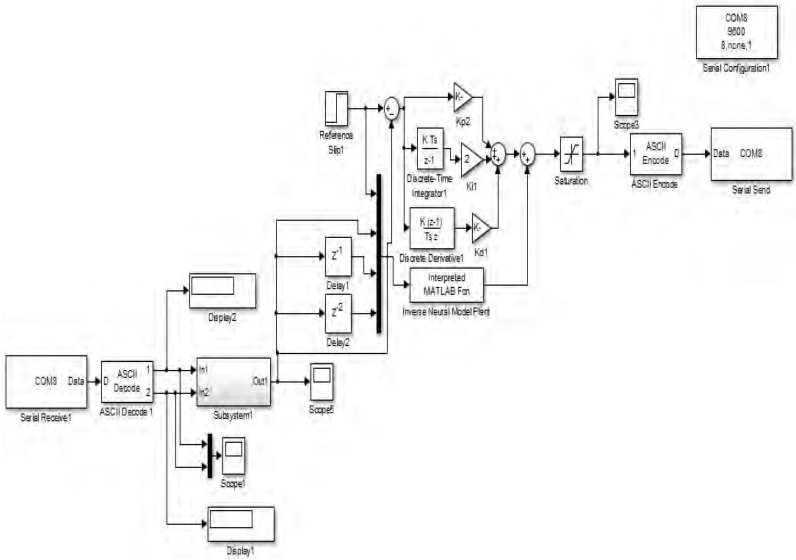
Gambar 3.10 Hasil Rancangan Program Sinyal PWM untuk Motor DC dan Rem Mikro Elektromagnetik

3.3.3 *Software* MATLAB

Selain *software* Arduino IDE, *software* lain yang digunakan pada Tugas Akhir ini adalah *software* MATLAB. *Software* ini digunakan untuk perancangan kontroler simulator ABS dengan menggunakan bahasa pemrograman MATLAB dan *toolbox* Simulink. Simulink digunakan untuk melakukan proses identifikasi dan permodelan *open loop* sistem dengan menggunakan *System Identification Toolbox*. Selain itu, Simulink juga digunakan untuk mendesain, mensimulasikan, dan mengimplementasikan metode kontrol yang digunakan pada simulator ABS. Pada Gambar 3.11 dapat dilihat diagram blok simulasi hasil perancangan kontroler yang digunakan dengan MATLAB Simulink. Sedangkan pada Gambar 3.12 dapat dilihat diagram blok implementasi perancangan kontroler yang digunakan dengan MATLAB Simulink.



Gambar 3.11. Diagram Blok Simulasi Perancangan Kontroler yang Digunakan dengan MATLAB Simulink

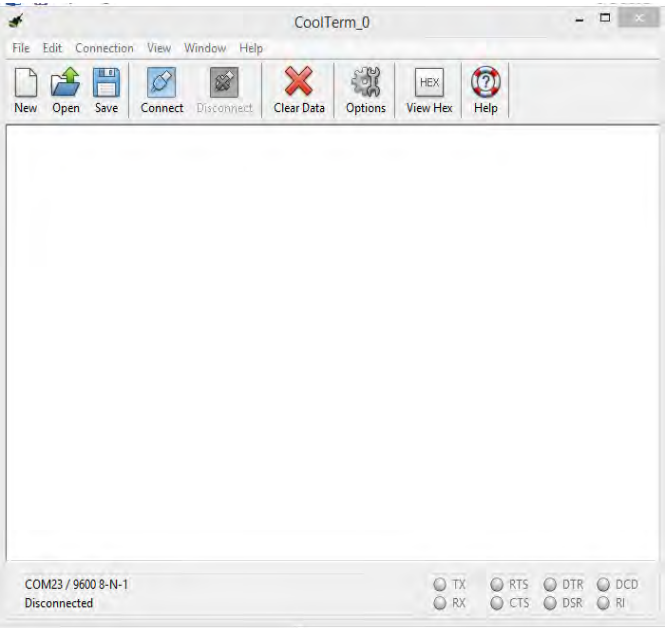


Gambar 3.12. Diagram Blok Implementasi Perancangan Kontroler yang Digunakan dengan MATLAB Simulink

3.3.4 *Software* CoolTerm

Software CoolTerm pada Tugas Akhir ini digunakan mengubah data kecepatan putar hasil pembacaan sensor *rotary encoder* oleh Arduino yang dikirimkan ke komputer dalam bentuk *file.txt* sehingga data lebih mudah untuk diolah. Software CoolTerm sendiri adalah salah satu aplikasi *serial terminal* yang membuat komunikasi di dalam mikrokontroler lebih mudah dilihat datanya. Melalui program ini, kita diijinkan untuk melihat data yang dikirimkan dari dan ke mikrokontroler, sensor, atau perangkat elektronik melalui *serial port* di layar monitor. Data yang dilihat tersebut dapat digunakan untuk beberapa kasus dalam rangkaian elektronik seperti *troubleshooting/debugging*, percobaan komunikasi antara dua rangkaian elektronik, kalibrasi hasil pembacaan sensor, dan *monitoring* data. *Software* ini juga menyediakan fasilitas untuk meng-*capture* data yang dikirimkan melalui *serial port* menjadi *textfile* dengan tambahan keterangan berupa kapan data diterima oleh *serial port*.

Software ini memerlukan sedikit pengaturan agar kita dapat menggunakannya. Hal pertama yang perlu diatur adalah pada *com* berapa sensor, perangkat elektronik atau mikrokontroler terhubung dengan *serial port*. Pengaturan tersebut dilakukan melalui *toolbox options* pada *bar serial port*. Selain itu, kita juga perlu mengatur *baud rate* data yang dikirimkan melalui *serial port*, sebab kesalahan dalam pemilihan baud rate menyebabkan data yang dikirimkan diartikan salah. Lalu pada *toolbox options* dan *bar receive*, pilihan *add timestamps to received data* untuk memberikan tambahan keterangan berupa kapan data diterima oleh *serial port* pada *file.txt*. Untuk melakukan *capture* terhadap data yang dikirimkan melalui *serial port*, kita dapat membuka *toolbar connection* dan memilih *Capture to Textfile Start.* Setelah itu kita beri nama pada *file* yang akan menyimpan hasil *capture* data yang dikirimkan dari sensor, rangkaian elektronik, dan mikrokontroler yang terhubung melalui *serial port*. Selanjutnya untuk menampilkan data yang dikirimkan melalui *serial port* di monitor kita harus menekan *toolbox connect*. Sedangkan *toolbox disconnect* digunakan untuk menghentikan program untuk menampilkan data yang dikirimkan melalui *serial port* dari perangkat elektronik di monitor. Jika kita ingin menghapus data komunikasi *serial port* yang sebelumnya sudah ditampilkan pada pada layar monitor, maka kita dapat menggunakan *toolbox Clear Data*. Tampilan utama dari *software* CoolTerm dapat dilihat pada Gambar 3.13.



Gambar 3.13. Tampilan Utama *Software* CoolTerm

3.4 Proses Identifikasi Sistem

Proses identifikasi bertujuan untuk mendapatkan model pendekatan sistem berdasarkan masukan dan keluaran yang diberikan kepada sistem. Pada Tugas Akhir ini, proses pemodelan simulator ABS menggunakan *AutoRegressive with eXternal input model estimaor* pada *System Identificaction Toolbox* MATLAB Simulink. Model pendekatan sistem yang didapatkan adalah model *Auto-Regressive* (AR) dengan metode estimasi parameter *least square*.

3.4.1 Pemilihan Kecepatan Awal Simulator ABS

Sebelum kita melakukan identifikasi *open loop*, kita harus menentukan kecepatan awal simulator sebelum melakukan pengereman. Pada Tabel 3.1, kita dapat melihat hubungan berbagai nilai sinyal PWM rem mikro elektromagnetik terhadap slip yang dihasilakan oleh roda atas dan roda bawah saat kecepatan awal simulator 2133 – 2400 rpm.

Tabel 3.1 Pengaruh Berbagai Nilai Sinyal PWM Rem Terhadap Slip saat Kecepatan Awal Simulator 2133 – 2400 RPM

Sinyal PWM Rem	Slip yang Dihasilkan Pada Kecepatan Awal Motor DC			
	2313 rpm	2220 rpm	2193 rpm	2155 rpm
10 %	0,12934	0,08476	0,16076	0,38118
20 %	0,14616	0,12543	0,17403	0,38464
30 %	0,17498	0,13771	0,21248	0,43337
40 %	0,23304	0,24226	0,28326	0,37228
50 %	0,48866	0,50346	0,44012	0,40046
60 %	0,66523	0,68928	0,66226	0,62183
70 %	0,76070	0,78347	0,74383	0,71080
80 %	0,82027	0,80893	0,79062	0,82936
90 %	0,81499	0,84319	0,87207	0,85293
10 %	0,83794	0,86017	0,86869	0,85408

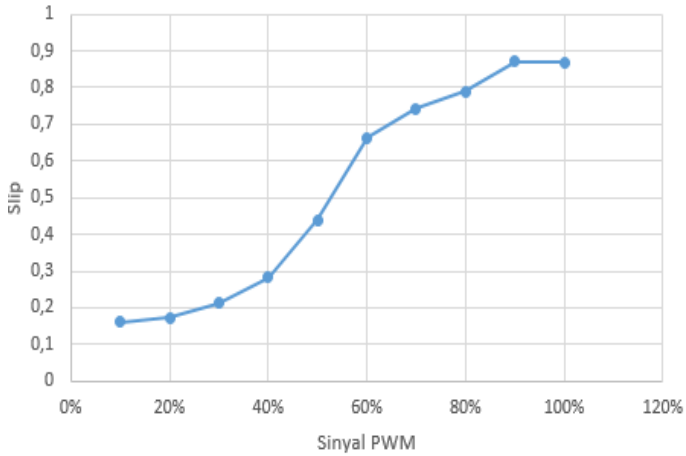
Dari data tersebut, dapat dilihat bahwa saat kecepatan awal simulator kurang dari 2155 rpm dan dilakukan pengereman dengan sinyal PWM 10 %, maka slip simulator telah melebihi nilai 0,2. Maka dari itu, kami kelompok ABS memutuskan untuk memilih kecepatan awal simulator ABS pada daerah dimana dengan sinyal PWM rem yang paling kecil (10%) slip yang dihasilkan simulator tidak melebihi nilai 0,2 yaitu pada kecepatan 2193 rpm – 2500 rpm. Pada Tugas Akhir ini, penulis memilih kecepatan awal simulator ABS sebelum melakukan proses pengereman adalah 2193 rpm.

3.4.2 Identifikasi *Open Loop* Simulator ABS pada Kecepatan Awal Simulator 2193 RPM

Identifikasi *open loop* simulator ABS dilakukan dengan memberikan sinyal uji *random* pada *plant* sehingga nantinya didapatkan hubungan *input output plant* yang menjadi data untuk dilakukan pemodelan sistem deterministik AR. Identifikasi dilakukan pada kecepatan awal simulator 2193 rpm dan sinyal PWM rem yang bervariasi mulai dari 10%, 20%, 30% sampai 100%. Besar slip yang dihasilkan simulator terhadap sinyal PWM rem yang diberikan pada saat kecepatan awal simulator 2193 rpm dapat dilihat pada Tabel 3.2 dan Gambar 3.14.

Tabel 3.2 Identifikasi *Open Loop* Simulator ABS pada Kecepatan Awal 2193 RPM

Nomor	Sinyal PWM Rem	Slip Roda Atas Terhadap Roda Bawah
1	10 %	0,160759
2	20 %	0,174033
3	30 %	0,212484
4	40 %	0,28316
5	50 %	0,440122
6	60 %	0,662262
7	70 %	0,743834
8	80 %	0,790617
9	90 %	0,872072
10	100 %	0,868692



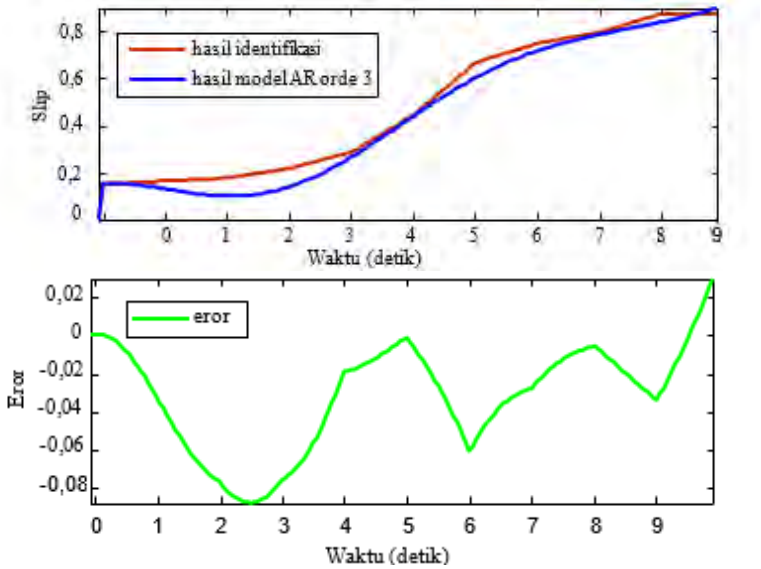
Gambar 3.14 Slip VS Berbagai Nilai Sinyal PWM Rem pada saat Kecepatan Awal Simulator 2193 RPM

3.4.3 Pemodelan dan Validasi Simulator ABS dengan Kecepatan Awal 2193 RPM

Setelah mendapatkan hubungan *input output* simulator ABS, proses identifikasi sistem selanjutnya adalah pemodelan *plant* representasi diskrit sistem deterministik AR dengan menggunakan *AutoRegressive with eXternal Input model estimator* pada *System Identification Toolbox* MATLAB Simulink.

Tabel 3.3 Validasi dan Pemodelan Simulator ABS pada Kecepatan Awal 2193 RPM dengan Berbagai Jumlah *na* Sistem Deterministik AR

Nomor	*na*	Fungsi Alih *Plant*	RMSE (%)
1	1	$\frac{Y(z)}{U(z)} = \frac{0{,}011545}{z - 0{,}99421}$	4,18493722
2	2	$\frac{Y(z)}{U(z)} = \frac{0{,}01325\ z}{z^2 - 1{,}1035z + 0{,}11173}$	1,282959
3	3	$\frac{Y(z)}{U(z)} = \frac{0{,}0018405\ z^2}{z^3 - 1{,}9811z^2 + 0{,}98103z + 0{,}0019625}$	0,183622



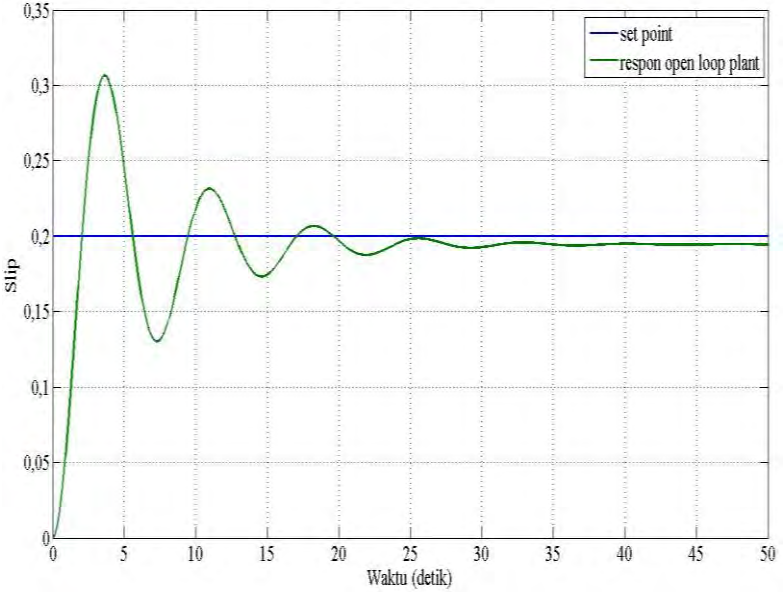
Gambar 3.15 Perbandingan Hasil Identifikasi VS Hasil Permodelan Sistem Deterministik AR Orde 3

Metode estimasi parameter yang digunakan *AutoRegressive with eXternal Input model estimator* pada *System Identification Toolbox* MATLAB Simulink adalah metode *least square*. Pada Tugas Akhir ini, juga digunakan metode *root mean square error* (RMSE) sebagai validasi pemodelan sistem deterministik AR yang didapatkan dari *System*

Identification Toolbox MATLAB Simulink dengan data identifikasi *open loop* simulator ABS. Berdasarkan Tabel 3.3 didapatkan model sistem deterministik AR dengan jumlah *na* 3 memiliki nilai RMSE yang kecil, sehingga dapat digunakan sebagai fungsi alih *plant* yang nantinya berguna untuk perancangan dan simulasi kontroler yang digunakan. Pada Gambar 3.15 juga dapat dilihat perbandingan hasil identifikasi *open loop* dengan hasil permodelan sistem deterministik AR dengan jumlah *na* 3.

3.5 Perancangan Kontroler PID

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 nilai masukan adalah sinyal PWM 20 % atau 0,2 dengan *setpoint* slip 0,2. Dapat dilihat pada Gambar 3.16 bahwa *respon open loop* (warna merah) tidak dapat mencapai *setpoint* (warna biru) dan respon sistem memiliki *overshoot*.



Gambar 3.16 Hasil Respon *Open Loop* Identifikasi Sistem

Berdasarkan Gambar 3.16 diperoleh spesifikasi respon *transient plant* sebagai berikut :

$$Xss = 0{,}2 \tag{3.3}$$

$$Yss = 0{,}1945 \tag{3.4}$$

$$K = \frac{Yss}{Xss} = \frac{0{,}1945}{0{,}2} = 0{,}9725 \tag{3.5}$$

$$\textit{Time constant } (\tau) = 1{,}5 \text{ detik} \tag{3.6}$$

$$Y(\tau) = 0{,}632 \text{ x } Yss = 0{,}632 \text{ x } 0{,}1945 = 0{,}122924 \tag{3.7}$$

$$\textit{Rise time } (t_r)\ 0 - 100\% = 2{,}05 \text{ detik} \tag{3.8}$$

$$\textit{Settling time } (t_s) \pm 0{,}5\% = 20{,}31 \text{ detik} \tag{3.9}$$

$$\textit{Delay time } (t_d) = 1{,}3 \text{ detik} \tag{3.10}$$

Sedangkan respon *steady-state* sistem adalah :

$$\%\, e_{ss} = \frac{Yss - Xss}{Xss} = \frac{0{,}2 - 0{,}1945}{0{,}2} \text{ x } 100\% = 2{,}75\ \% \tag{3.11}$$

$$\text{Maksimum } \textit{overshoot} = \frac{0{,}3064 - 0{,}1945}{0{,}1945} = 0{,}5753 = 57{,}53\ \% \tag{3.12}$$

Dari respon *open loop plant* tersebut maka dirancanglah parameter kontroler PID dengan metode *cut and paste*. Metode *cut and paste* dilakukan karena fungsi alih *plant* yang didapatkan termasuk fungsi alih orde tinggi sehingga tidak mudah untuk menghitung parameter kontroler PID secara analitik. Metode *cut and paste* adalah metode perancangan parameter kontroler PID dengan cara memilih berbagai jenis mode kontroler PID yang sesuai dengan respon *plant* yang diinginkan. Pada saat perancangan kontroler PID spesifikasi yang diinginkan adalah *rise time* kurang dari sama dengan 1 detik, s*ettling time* ±0,5% kurang dari sama dengan 15 detik, dan respon *plant* tidak memiliki *error steady-state* dan *overshoot*.

Langkah – langkah dalam melakukan metode *cut and paste* adalah sebagai berikut :

- Pertama – tama penulis memasang kontroler *propotional* terlebih dahulu, dan mengabaikan kontroler *integratif* dan

derivatif dengan memberikan nilai nol pada kedua konstantanya.

- Lalu penulis terus menambahkan konstanta *propotional* sampai *rise time plant* menjadi 1 detik.
- Untuk meredam *ovesrshoot* dan osilasi, penulis menambahkan kontroler *derivatif* dimana nilai konstantanya setengah nilai konstanta *propotional*, sambil mengamati keadaan respon *plant* hingga stabil dan lebih responsif.
- Jika respon *plant* masih memiliki *error steady-state* yang tinggi, penulis menambahkan kontroler *integratif* dengan memperhatikan respon *plant*.
- Lalu penulis memeriksa kembali respon *plant* dengan nilai parameter yang telah didapatkan sampai mendapatkan hasil respon *plant* yang memuaskan.

Berdasarkan langkah – langkah metode *cut and paste* diatas didapatkan nilai dari K_p, K_i dan K_d adalah 8, 2, 10.

3.6 Perancangan Kontroler Jaringan Saraf Tiruan Berbasis *Feedforward* yang Mengoptimalkan PID

Jaringan saraf tiruan jika digunakan sebagai kontroler maka jaringan saraf tersebut menggunakan *inverse* dari proses *plant* sebagai proses pembelajaran. Salah satu jenis metode kontroler yang menggunakan *inverse* dari proses *plant* adalah jaringan saraf tiruan yang mengoptimalkan kontroler PID, seperti yang dapat dilihat pada Gambar 2.17. Metode kontrol tersebut, menggunakan jaringan saraf tiruan yang merupakan *inverse neural model plant* sebagai kontroler *feedforward* untuk menghasilkan sinyal *feedforward* ke dalam sinyal kontrol ke *plant*, sehingga diharapkan dapat memperbaiki respon *transient plant* yang tidak dapat diperbaiki oleh kontroler PID.

3.6.1 *Inverse Model* Simulator ABS

Sebelum kita menentukan data *input* dan target pembelajaran jaringan saraf tiruan, maka kita harus mengetahui dahulu persamaan *inverse model* simulator ABS. Persamaan *invers model* simulator ABS diperoleh dari persamaan representasi diskrit sistem deterministik AR orde 3 yang didapatkan pada proses identifikasi sebelumnya. Persamaan representasi diskrit sistem deterministik AR orde 3 itu dapat dilihat lagi pada Persamaan (3.13).

$$\frac{Y(z)}{U(z)} = \frac{0{,}0018405\, z^2}{z^3 - 1{,}9811\, z^2 + 0{,}98103\, z + 0{,}0019625} \quad (3.13)$$

Persamaan representasi diskrit sistem deterministik AR orde 3 pada Persamaan (3.13) tersebut dapat diubah menjadi persamaan beda yaitu:

$$y(k) = 1{,}9811y(k\text{-}1) - 0{,}98103y(k\text{-}2) - 0{,}001963y(k\text{-}3) + 0{,}0018405\, u(k\text{-}1) \quad (3.14)$$

Berdasarkan Persamaan (3.14) kita dapat memperoleh persamaan prediksi simulator ABS dengan menganggap *output* sekarang sebagai *output* masa depan dan *input* masa lalu menjadi *input* sekarang, yaitu:

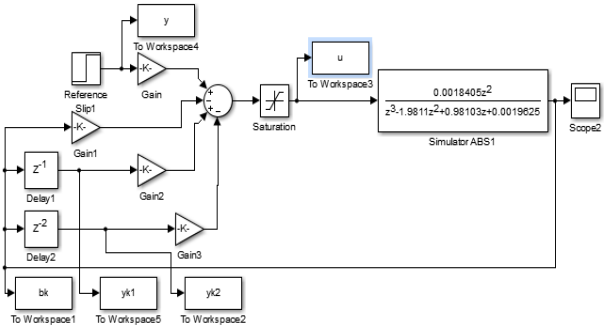
$$y(k+1) = 1{,}9811y(k) - 0{,}98103y(k\text{-}1) - 0{,}001963\, y(k\text{-}2) + 0{,}0018405\, u(k) \quad (3.15)$$

Berdasarkan Persamaan (3.15) kita dapat memperoleh persamaan *invers model* simulator ABS dengan menukar posisi variabel $u(k)$ dengan variabel $y(k)$, yaitu:

$$u(k) = 543{,}330617y(k+1) - 1076{,}39228y(k) + 533{,}023635\, y(k\text{-}1) + 1{,}06628634\, y(k\text{-}2) \quad (3.16)$$

3.6.2 *Input* dan Target Pembelajaran

Sebelum kita melakukan proses pembelajaran, kita terlebih dahulu harus menentukan data *input* dan target pembelajaran. Hal ini penting dilakukan agar jaringan saraf tiruan yang akan digunakan sesuai dengan yang kita inginkan. Pada Tugas Akhir ini, proses untuk mendapatkan data *input* dan target pembelajaran melalui *software* MATLAB Simulink dapat dilihat pada Gambar 3.17.



Gambar 3.17 Diagram Blok Pengambilan Data *Input* dan Target Pembelajaran dengan MATLAB Simulink

Pada *inverse neural model plant* yang menjadi masukannya adalah slip simulator ABS, sedangkan yang menjadi target pembelajarannya adalah sinyal PWM rem. Berdasarkan Gambar 3.18, yang menjadi data target pembelajaran *inverse neural model plant* adalah *inverse model plant* simulator ABS yang mengalami saturasi dengan nilai batas atas 1, nilai batas bawah 0, dan waktu *sampling* 0,03. Pemilihan nilai saturasi tersebut berdasarkan pada *range* kerja sinyal kontrol sesungguhnya yang digunakan simulator ABS. Sedangkan berdasarkan Gambar 3.18, yang menjadi data *input inverse neural model plant* adalah slip simulator ABS yang nilainya digeser ($y(k+1)$, $y(k)$, $y(k-1)$, dan $y(k-2)$). Banyaknya data *input* dan target pembelajaran yang digunakan pada Tugas Akhir ini adalah 134. Penentuan banyaknya data yang akan digunakan dapat dilakukan secara bebas dan tidak ada ketentuan terkait yang mengaturnya.

3.6.3 Fungsi Aktivasi

Fungsi aktivasi adalah elemen pemprosesan yang terdapat pada setiap *neuron* kecuali *neuron* pada lapisan *input*, yang menyatakan kondisi keluaran dari setiap *neuron* pada jaringan saraf tiruan. Fungsi aktivasi yang digunakan pada Tugas Akhir ini adalah fungsi *sigmoid unipolar* pada lapisan tersembunyi dan pada lapisan *output*. Persamaan matematika dari fungsi aktivasi tersebut dapat dilihat pada Tabel 3.4[16].

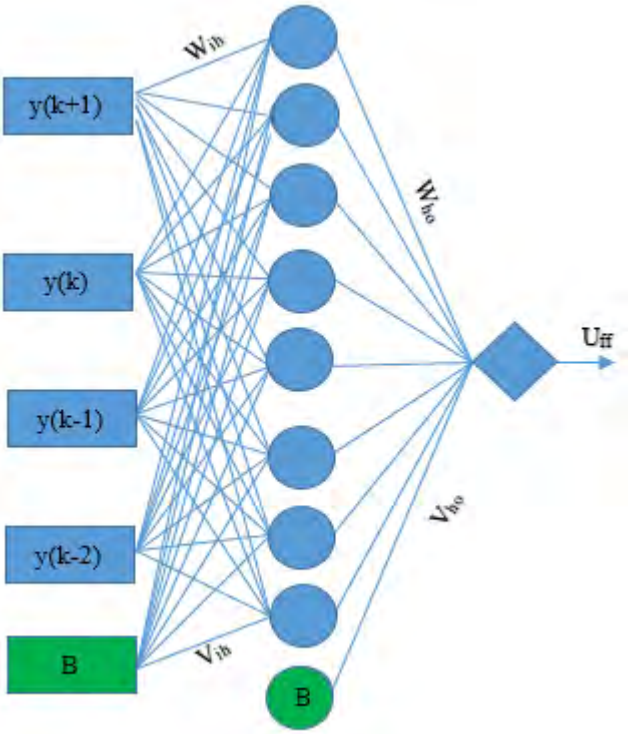
Tabel 3.4 Persamaan Matematika Fungsi Aktivasi yang Digunakan

Fungsi *Sigmoid Unipolar*	$y = \frac{1}{1 + e^{-x}}$

Fungsi tersebut dipilih karena fungsi aktivasi tersebut memiliki karakteristik yaitu dapat bernilai pada setiap titik, dapat menerima *input* sampai nilainya tidak terhingga tetapi dapat menghasilkan *output* pada kawasan terhingga, serta memiki pola yang sama dengan fungsi aktivasi jaringan saraf otak.

3.6.4 Arsitektur Jaringan Saraf Tiruan

Struktur jaringan saraf tiruan yang digunakan pada Tugas Akhir ini adalah struktur *feedforward*, sehingga jaringan saraf yang digunakan memiliki arsitektur yang terdiri dari tiga lapisan yaitu lapisan *input*, tersembunyi, dan *output* dengan masing – masing *neuron* di dalamnya.



Gambar 3.18 Arsitektur *Inverse Neural Model Plant*

Pada lapisan *input*, jumlah *neuron* yang digunakan adalah 4 mengikuti jumlah masukan dari *inverse model plant* pada Persamaan (3.13), sedangkan pada lapisan tersembunyi jumlah *neuron* yang digunakan adalah 8. Pada lapisan *output* jumlah *neuron* yang digunakan adalah 1 mengikuti jumlah keluaran *inverse model plant* simulator ABS (sinyal $u(k)$). Penentuan arsitektur jaringan saraf tiruan pada Tugas Akhir ini ditentukan dari hasil pembelajaran dengan *Mean Square Error* (MSE) terkecil yang dapat dilihat pada Tabel 3.5. Arsitektur jaringan saraf tiruan yang digunakan pada Tugas Akhir ini dapat dilihat pada Gambar 3.18.

Pada Gambar 3.18 dapat dilihat bahwa simbol yang berwarna hijau mewakili bias dan selalu bernilai 1, sedangkan simbol yang berwarna biru mewakili *neuron* pada arsitektur *inverse neural model plant* simulator

ABS. Pada Gambar 3.18 simbol persegi panjang mewakili *neuron* pada lapisan *input,* sedangkan simbol lingkaran mewakili *neuron* dan bias pada lapisan tersembunyi. Simbol belah ketupat pada Gambar 3.18 juga mewakili *neuron* dan bias pada lapisan *output*. Masing - masing *neuron* pada Gambar 3.18 saling terhubung dengan neuron yang lain dengan bobot tertentu. *Neuron* pada lapisan *input* terhubung dengan *neuron* pada lapisan tersembunyi dengan bobot penghubung W_{ih}, dimana indeks *i* mewakili *neuron* ke-*i* pada lapisan *input*, sedangkan indeks h mewakili *neuron* ke-*h* pada lapisan tersembunyi. Begitu juga dengan *neuron* pada lapisan tersembunyi, *neuron* pada lapisan tersebut juga terhubung dengan *neuron* pada lapisan *output* dengan bobot penghubung W_{ho}, dimana indeks h mewakili *neuron* ke-*h* pada lapisan tersembunyi, sedangkan indeks *o* mewakili *neuron* ke-*o* pada lapisan *output*. Bias pada arsitektur *inverse neural model plant* simulator ABS juga terhubung *neuron – neuron* yang ada, seperti bias pada lapisan *input* terhubung dengan *neuron* pada lapisan tersembunyi dengan bobot penghubung V_{ih}, dimana indeks *i* mewakili bias ke-*i* pada lapisan *input*, sedangkan indeks *h* mewakili *neuron* ke-*h* pada lapisan tersembunyi. Bias pada lapisan tersembunyi dengan *neuron* pada lapisan *output* juga saling terhubung dengan bobot penghubung V_{ho}, dimana indeks *h* mewakili bias ke-*h* pada lapisan tersembunyi, sedangkan indeks *o* mewakili *neuron* ke-*o* pada lapisan *output.*

3.6.5 Pembelajaran *Inverse Neural Model Plant*

Pada Tugas Akhir ini, proses pembelajaran jaringan saraf tiruan menggunakan struktur *generalized architecture* sehingga dapat dilakukan secara *offline*. Proses pembelajaran dengan struktur ini memberikan sinyal *u* kepada *plant* sehingga akhirnya *plant* akan menghasilkan *output* yang berupa sinyal *y*. *Output* dari *plant* akan masuk ke dalam jaringan saraf tiruan sebagai masukan. Lalu jaringan saraf akan memberi tanggapan berupa sinyal *Un*. Nantinya sinyal *Un* akan dibandingkan dengan sinyal *u* yang masuk ke dalam *plant*, jika masih terdapat *error* maka akan digunakan untuk memperbaiki bobot jaringan, jika terdapat *error* maka jaringan saraf tiruan tersebut sudah bisa meniru *inverse* dari *plant*. Struktur pembelajaran *generalized architecture* dapat dilihat pada Gambar 2.9.

Pada Tugas Akhir ini, algoritma pembelajaran jaringan saraf tiruan yang digunakan adalah *backpropagation.* Dalam metode ini, proses pembelajaran dilakukan dengan melihat *error* antara keluaran jaringan saraf tiruan dengan target pembelajaran. Selama *error* yang dihasilkan

melebihi batas tertentu, maka akan dilakukan perubahan bobot pada masing – masing sampai *error* terhadap target pembelajaran tidak melebihi batas tertentu. Proses pembelajaran *inverse neural model* yang dilakukan pada Tugas Akhir ini adalah sebagai berikut:

- Menentukan jumlah lapisan dan *neuron* yang akan digunakan pada jaringan saraf tiruan, jumlah *epoch, stop criteria*, dan laju pembelajaran. Nilai laju pembelajaran terletak antara 0 dan 1, semakin besar nilai laju pembelajaran *epoch* yang dipakai semakin sedikit, tetapi *error* yang dihasilkan tidak stabil. Jika laju pembelajaran yang dipilih nilainya kecil maka *error* akan konvergen dalam waktu yang sangat lama.
- Menginisialisasi bobot – bobot *neuron* dan bias yang akan digunakan dalam jaringan saraf tiruan dengan nilai *random* kecil antara -0,5 sampai 0,5, karena bobot awal sangat memengaruhi seberapa cepat *error* yang dihasilkan jaringan konvergen.
- Menerima keluaran dari *plant* sebagai masukan pembelajaran, dan masukan plant sebagai target pembelajaran.
- Menghitung semua keluaran di unit tersembunyi (*yh*)

$$z\_h = V_{ih} + \sum_{i=0}^{n} x_n * W_{ih_n} \tag{3.17}$$

$$yh = f_{\text{sigmoid unipolar}}(z\_h) \tag{3.18}$$

- Menghitung keluaran di unit *output* (*yo*)

$$z\_o = V_{oh} + \sum_{i=0}^{m} yh_m * W_{oh_n} \tag{3.19}$$

$$yo = f_{\text{sigmoid unipolar}}(z\_o) \tag{3.20}$$

- Menghitung *error* (e_{out}) dan *delta error* lapisan *output* (*delo*) yang terjadi pada lapisan *output*

$$e_{out} = \textit{target pembelajaran} - yo \tag{3.21}$$

$$delo = e_{out} * yo * (1 - yo) \tag{3.22}$$

- Menghitung MSE dan membandingkan dengan *stop criteria*

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(e_{out})^2 \tag{3.23}$$

- Jika MSE > *stop criteria*, maka akan dilakukan penghitungan perambatan *error* (e_h) dan *delta error* lapisan tersembunyi (*delh*). Jika MSE > *stop criteria*, maka proses pembelajaran berhenti. Akan tetapi jika MSE > *stop criteria* dan jumlah *epoch* yang ditentukan sudah habis maka proses pembelajaran juga berhenti.

$$e_h = delo * W_{oh_n} \tag{3.24}$$

$$delh = eh * y_{h_n} * (1 - y_{h_n}) \tag{3.25}$$

- Menghitung bobot *neuron* dan bias baru pada lapisan tersembunyi ke lapisan *output*.

$$W_{oh_{baru}} = W_{oh_{lama}} + delo * \propto * yh_n \tag{3.26}$$

$$V_{oh_{baru}} = V_{oh_{lama}} + delo * \propto \tag{3.27}$$

- Menghitung bobot *neuron* dan bias baru pada lapisan *input* ke lapisan tersembunyi.

$$W_{ih_{baru}} = W_{ih_{lama}} + delh * \propto * x_n \tag{3.28}$$

$$V_{ih_{baru}} = V_{ih_{lama}} + delh * \propto \tag{3.29}$$

- Kembali ke proses awal

Proses pembelajaran *inverse neural model plant* menggunakan data sebanyak 134 data tiap iterasi. Lamanya kondisi pembelajaran ditentukan berdasarkan junlah *neuron* pada lapisan tersembunyi, laju pembelajaran yang digunakan, dan minimum *error* yang diinginkan. Pada Tugas Akhir ini, nilai minimum *error* didapatkan dengan persamaan MSE dengan *error* yang diharapkan 0,00001, dan jumlah *epoch* yang digunakan adalah 100000.

Penentuan arsitektur awal *inverse neural model plant* simulator ABS dilakukan dengan *trial and error*. Jumlah *input* dan *output node* disesuaikan dengan Persamaan (3.13). Sedangkan penetuan jumlah *neuron* pada lapisan tersembunyi berdasarkan nilai *mean square error* terkecil yang dapat dilihat pada Tabel 3.5.

Tabel 3.5 Variasi Perubahan Jumlah *Neuron* pada Lapisan Tersembunyi dan Laju Pembelajaran Terhadap M*ean Square Error* yang Dihasilkan *Inverse Neural Model Plant*

No.	*Neuron* pada *Hidden Layer*	Laju Pembelajaran	MSE
1	1	0,005	0,0094
2	8	0,005	0,0093
3	16	0,005	0,0087
4	1	0,1	0,0092
5	8	0,1	0,0078
6	16	0,1	0,0064
7	1	0,2	0,0063
8	8	0,2	0,0061
9	16	0,2	0,0057
10	1	0,5	0,0051
11	8	0,5	0,0042
12	16	0,5	0,0046
13	1	1	0,0074
14	8	1	0,0028
15	16	1	0,0028

Berdasarkan Tabel 3.5, jumlah 8 *neuron* pada lapisan tersembuyi dengan nilai laju pembelajaran 1 menghasilkan nilai MSE terkecil, sehingga jumlah *neuron* pada lapisan tersembunyi tersebut digunakan sebagai arsitektur jaringan saraf tiruan. Maka dapat disimpulkan bahwa dalam pembelajaran jaringan saraf tiruan yang merupakan *inverse neural model plant* ini digunakan empat neuron pada lapisan *input*, 8 *neuron* pada lapisan tersembunyi, dan satu *neuron* pada lapisan *output.*

3.6.6 Proses Pembentukan Sinyal Kontrol *Feedforward*

Proses pembentukan sinyal kontrol *feedforward* (U_{ff}) pada metode kontrol jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID sebenarnya hampir sama dengan proses pembelajaran *inverse neural model plant,* yang membedakan hanya bobot yang digunakan dan proses yang dilakukan. Pada proses pembentukan sinyal kontrol U_{ff} bobot awal yang digunakan bukan lagi nilai *random* dengan rentang - 0,5 sampai 0,5, melainkan bobot saat *epoch* terakhir (*epoch* ke 100.000) proses pembelajaran *inverse neural model plant.* Bobot yang digunakan pada algoritma pembentukan sinyal kontrol *feedforward* adalah

- Bobot penghubung *neuron* pada lapisan *input* dengan *neuron* pada lapisan tersembunyi (W_{ih})
  $W_{i,1}$ = [-0.1664 -28,5693 -3,94958 14,3810]
  $W_{i,2}$ =[-2,94172 -99,426109 31,7115 128,0843]
  $W_{i,3}$ = [-0,8301 1.70851 -0.65039 1,99136]
  $W_{i,4}$ = [4,66065 -158,3008 -37,4081 56,1815]
  $W_{i,5}$ = [-6,28234 -108,67566 -38,2642 -11,331]
  $W_{i,6}$ = [-2,5022 91,374088 -14,14256 -40,5939]
  $W_{i,7}$ = [-0,8904 68,56507 2,110954 -45,2719]
  $W_{i,8}$ = [0,70662 47,464255 -6,47944 24,3257]
- Bobot penghubung *neuron* pada lapisan tersembunyi dengan *neuron* pada lapisan *output* (W_{ho})
  $W_{h,1}$= [-7,145; 12,6739; -1,5508; 40,4834; 33,38775; -17,522; 13,94594; -9,6822]
- Bobot penghubung bias pada lapisan *input* dengan *neuron* pada lapisan tersembunyi (V_{ih})
  $V_{1,h}$ = [1,176528; -12,84793; -3,2692; 24,1463; -28,8842; -14,539; 6,74038; -3,5255]
- Bobot penghubung bias pada lapisan tersembunyi dengan *neuron* pada lapisan *output* (V_{ho})
  $V_{1,1}$ = [-2,473539295830077]

Proses pembentukan sinyal kontrol U_{ff} juga hanya melakukan proses *feedforward* dan tidak melakukan proses *backward* lagi. Setelah sinyal kontrol U_{ff} terbentuk, selanjutnya sinyal kontrol U_{ff} akan digabungkan dengan sinyal kontrol PID untuk meningkatkan respon *transient plant*.

BAB 4
PENGUJIAN DAN ANALISIS SISTEM

Pada bab ini akan dilakukan pengujian dan analisis sistem. Pengujian yang dilakukan trerdiri dari dua jenis yaitu pengujian berupa simulasi dan pengujian berupa implementasi. Pengujian ini dilakukan untuk mengetahui seberapa bagus kontroler yang sudah dirancang dalam meregulasi slip simulator ABS menjadi 0,2.

4.1 Gambaran Umum Pengujian Sistem

Pada Tugas Akhir kali ini dilakukan pengujian sistem kontrol slip pada simulator ABS melalui simulasi dan implementasi. Dalam melakukan simulasi aspek pengujian yang dilakukan adalah kontroler yang telah dirancang yaitu kontroler PID, kontroler *inverse neural model plant,* dan kontroler jaringan saraf tiruan yang mengoptimalkan PID harus dapat meregulasi slip simulator ABS menjadi 0,2.

Sedangkan implementasi dilakukan untuk menguji performa kontroler yang telah dirancang ketika diterapkan langsung pada simulator ABS. Sinyal *input* yang diberikan kepada plant berupa sinyal PWM Arduino yang merepresentasikan torsi pengereman. Sedangkan o*utput* dari *plant* yang berupa slip antara roda atas dan roda bawah didapatkan dari pengolahan sensor *rotary encoder* di dalam Arduino. Aspek pengujian yang dilakukan dalam implementasi sama dengan aspek pengujian yang dilakukan dalam simulasi. Akan tetapi pada pengujian berupa implementasi kontroler yang digunakan dalam meregulasi slip simulator ABS hanyalah kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID.

4.2 Simulasi Sistem

Simulasi hasil perancangan kontroler terhadap simulator ABS dilakukan melalui *software* MATLAB Simulink. Simulasi ini bertujuan untuk mengetahui seberapa bagus kontroler yang telah dirancang dalam menyelesaikan permasalahan regulator slip pada simulator ABS. Simulasi dilakukan dengan membandingkan respon plant yang menggunakan kontroler jaringan saraf tiruan yang mengoptimalkan PID dengan kontroler PID, dan *inverse neural model plant*. Pengujian dilakukan dengan memberikan *set point* berupa sinyal *step* dengan nilai 0,2 dan waktu *sampling* 0,03.

4.2.1 Pengujian Menggunakan Kontroler PID

Pengujian ini dilakukan dengan menggunakan nilai K_p, K_i, dan K_d yang sebelumnya sudah didapatkan dengan metode *cut and paste* yaitu 8, 2, dan 10. Berdasarkan metode *cut and paste* untuk perancangan kontroler PID, maka respon *plant* dengan kontroler PID dapat dilihat pada Gambar 4.1, sedangkan sinyal kontrol yang dihasilkan dapat dilihat pada Gambar 4.2.

Berdasarkan Gambar 4.1 diperoleh spesifikasi respon *transient plant* dengan kontroler PID sebagai berikut :

$Yss = 0{,}2$ (4.1)

Time constant $(\tau) = 0{,}06$ detik (4.2)

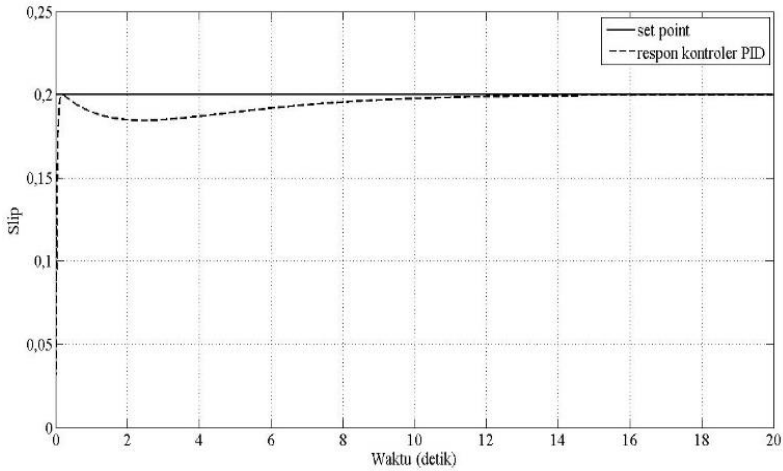
Rise time (t_r) 10-90% = 0,09 detik (4.3)

Settling time (t_s) ±0,5% = 12,35 detik (4.4)

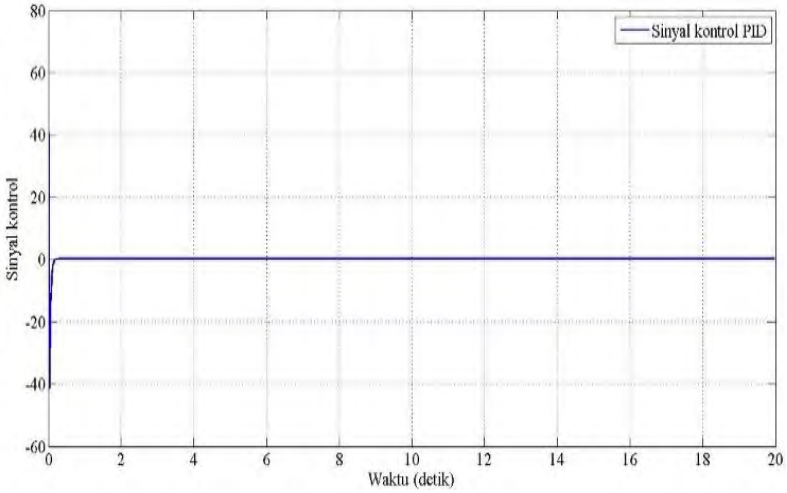
Sedangkan respon *steady-state* sistem adalah :

$$\%\, e_{ss} = \frac{Yss - Xss}{Xss} = \frac{0{,}2 - 0{,}2}{0{,}2} \text{x } 100\% = 0\ \% \quad (4.5)$$

$$\text{Maksimum } overshoot = \frac{0{,}2 - 0{,}2}{0{,}2} = 0 = 0\ \% \quad (4.6)$$



Gambar 4.1 Respon *Plant* dengan Kontroler PID

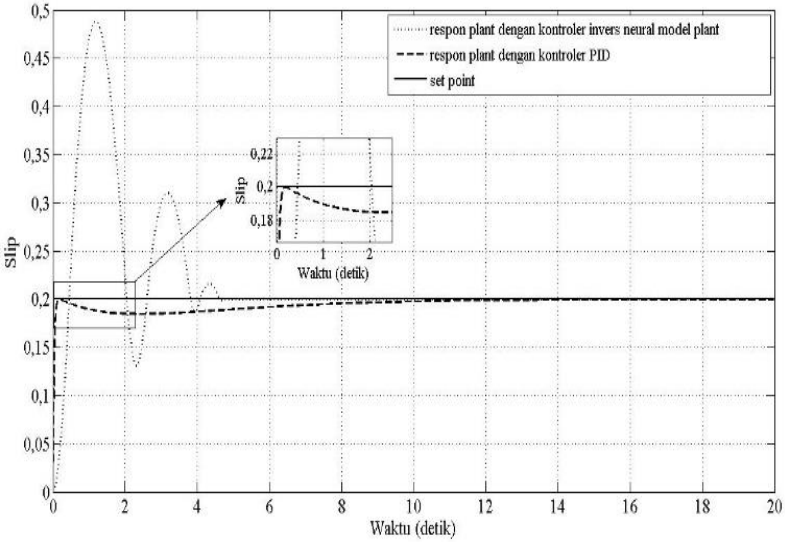


Gambar 4.2 Sinyal Kontrol yang Diberikan Kontroler PID kepada *Plant*

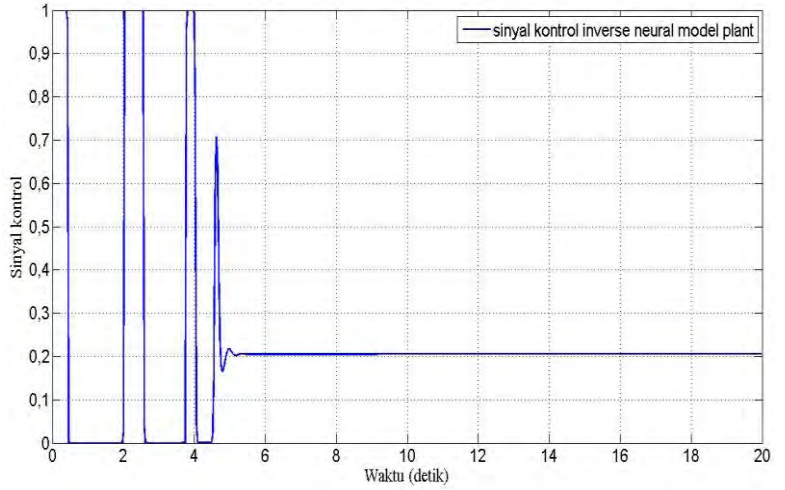
Berdasarkan Gambar 4.1 dapat dilihat bahwa respon *plant* dengan tidak memiliki *overshoot* dan baru mencapai *steady-state* setelah 16 detik. Pada Gambar 4.2 juga dapat dilihat bahwa sinyal kontrol yang diberikan kontroler PID bernilai dari – 40 sampai 40. Sinyal kontrol ini ternyata telah melebihi *range* kerja sinyal PWM rem yang bernilai dari 0 – 1. Akan tetapi hal ini dapat diatasi dengan meletakan blok *saturation* sebelum blok simulator ABS di *software* MATLAB Simulink. Blok ini berfungsi agar sinyal kontrol yang diberikan kontroler PID selalu bernilai dari 0 – 1, walaupun merubah sedikit hasil respon *plant* dengan kontroler PID.

4.2.2 Pengujian Menggunakan Kontroler *Inverse Neural Model Plant*

Pengujian ini dilakukan untuk mengetahui seberapa baik kontroler *inverse neural model plant* jika dibandingkan dengan kontroler PID yang telah dilakukan sebelumnya. Masukan dari kontroler *inverse neural model plant* adalah slip simulator ABS yang nilainya digeser ($y(k+1)$, $y(k)$, $y(k-1)$, dan $y(k-2)$). Sedangkan keluaran dari kontroler *inverse neural model plant* adalah sinyal PWM ($u(k)$). Berdasarkan bobot – bobot yang telah didapatkan dari proses pembelajaran, maka respon plant dengan kontroler *inverse neural model plant* dapat dilihat pada Gambar 4.3, sedangkan sinyal kontrol yang dihasilkan dapat dilihat pada Gambar 4.4.



Gambar 4.3 Respon *Plant* dengan Kontroler *Inverse Neural Model Plant*



Gambar 4.4 Sinyal kontrol yang diberikan Kontroler *Inverse Neural Model Plant* kepada *Plant*

Berdasarkan Gambar 4.3 diperoleh spesifikasi respon *transient plant* dengan kontroler *inverse neural model plant* sebagai berikut :

$Yss = 0{,}2$ (4.7)

Time constant (τ) = 0,36 detik (4.8)

Rise time (t_r) 0-100% = 0,48 detik (4.9)

Settling time (t_s) ±0,5 % = 4,7 detik (4.10)

Sedangkan respon *steady-state* sistem adalah :

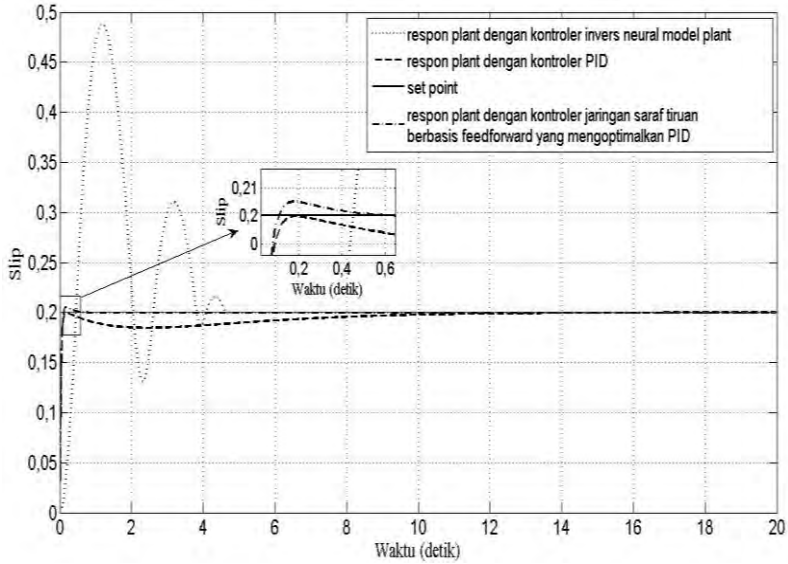
$$\% \, e_{ss} = \frac{Yss - Xss}{Xss} = \frac{0{,}2 - 0{,}2}{0{,}2} \text{ x } 100\% = 0 \ \% \quad (4.11)$$

$$\text{Maksimum } overshoot = \frac{0{,}5 - 0{,}2}{0{,}2} = 0 = 0 \ \% \quad (4.12)$$

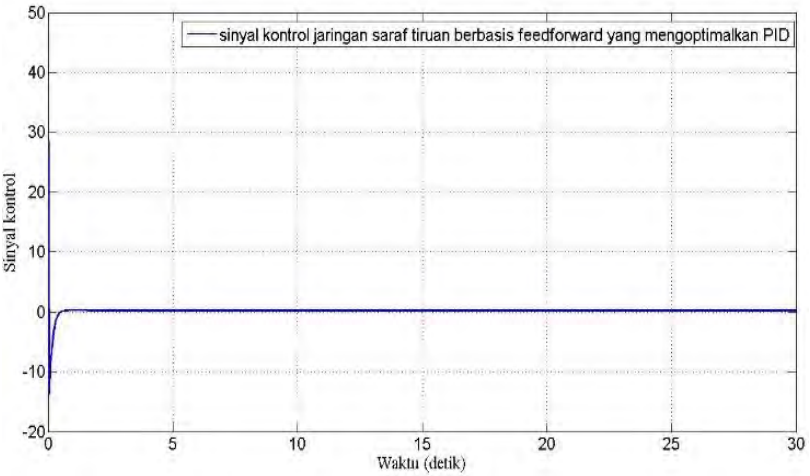
Berdasarkan Gambar 4.3 dapat dilihat bahwa respon *plant* dengan kontroler *inverse neural model plant* lebih cepat mencapai *steady-state* jika dibandingkan dengan kontroler PID yaitu setelah 5 detik. Akan tetapi respon *plant* dengan kontroler *inverse neural model plant* memiliki *overshoot* yang besar yaitu 0,5 dan sangat membahayakan pengemudi jika kontroler ini diterapkan pada ABS. Berdasarkan Gambar 4.4 dapat dilihat bahwa sinyal kontrol dari kontroler *inverse neural model plant* bernilai dari 0 – 1 dan telah sesuai dengan *range* kerja sinyal PWM rem.

4.2.3 Pengujian Menggunakan Kontroler Jaringan Saraf Tiruan Bebasis *Feedforward* yang Mengoptimalkan PID

Berdasarkan dua pengujian yang telah dilakukan sebelumnya, penulis mencoba untuk menggabungkan kedua kontroler tersebut agar keunggulan yang dimiliki kontroler *inverse neural model plant* dapat memperbaiki kekurangan yang dimiliki oleh kontroler PID dan begitu juga sebaliknya. Pengujian ini dilakukan untuk mengetahui seberapa baik hasil perancangan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID jika dibandingkan dengan kontroler PID dan kontroler *inverse neural model plant*. Nilai K_p, K_i, dan K_d yang digunakan sama dengan pengujian sebelumnya. Sedangkan bobot dan arsitektur jaringan saraf tiruan yang digunakan sama dengan pengujian sebelumnya. Respon plant dengan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID dapat dilihat pada Gambar 4.5, sedangkan sinyal kontrol yang dihasilkan dapat dilihat pada Gambar 4.6.



Gambar 4.5 Respon *Plant* dengan Kontroler Jaringan Saraf Tiruan Berbasis *Feedforward* yang Mengoptimalkan PID



Gambar 4.6 Sinyal Kontrol Jaringan Saraf Tiruan Berbasis *Feedforward* yang Mengoptimalkan PID kepada *Plant*

Berdasarkan Gambar 4.5 diperoleh spesifikasi respon *transient plant* dengan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID sebagai berikut :

$Yss = 0{,}2$ (4.13)

Time constant $(\tau) = 0{,}1$ detik (4.14)

Rise time (t_r) 0 - 100% = 0,3 detik (4.15)

Settling time $(t_s) \pm 0{,}5\% = 1{,}25$ detik (4.16)

Sedangkan respon *steady–state* sistem adalah :

$$\% \, e_{ss} = \frac{Yss - Xss}{Xss} = \frac{0{,}2 - 0{,}2}{0{,}2} \text{ x } 100\% = 0 \ \% \quad (4.17)$$

$$\text{Maksimum } overshoot = \frac{0{,}20153 - 0{,}2}{0{,}2} = 0{,}0765 = 7{,}65 \ \% \quad (4.18)$$

Berdasarkan Gambar 4.5 dapat dilihat bahwa respon *plant* dengan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID lebih cepat mencapai *steady-state* jika dibandingkan dengan kontroler PID dan kontroler *inverse neural model plant* yaitu setelah 1 detik. *Overshoot* yang diakibatkan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID sangat kecil yaitu 0,20153 jika dibandingkan dengan kontroler *inverse neural model plant*, walaupun tidak dapat seperti kontroler PID yang tidak memiliki *overshoot.* Akan tetapi *overshoot* yang dihasilkan masih dalam batas aman, karena ketika slip bernilai 0,3 koefisien gesek jalannya masih besar dan daya manuver mobil berada pada titik kritis sebelum mobil susah untuk dikendalikan. Pada Gambar 4.6 juga dapat dilihat bahwa sinyal kontrol dari kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID bernilai dari – 12 sampai 39. Sinyal kontrol ini ternyata telah melebihi *range* kerja sinyal PWM rem yang bernilai dari 0 – 1. Akan tetapi hal ini dapat diatasi dengan meletakan blok *saturation* sebelum blok simulator ABS di *software* MATLAB Simulink.

Untuk mengetahui masing – masing keunggulan dan kelemahan dari ketiga kontroler yang digunakan dalam pengujian berupa simulasi ini, maka kita dapat membandingkan respon *transient* dan respon *steady-state* masing – masing kontroler. Perbandingan respon *transient* dan respon *steady-state* respon *plant* pada saat simulasi yang menggunakan ketiga jenis kontroler dapat dilihat pada Tabel 4.1.

Tabel 4.1 Perbandingan Respon *Plant* dengan Berbagai Jenis Kontroler

Respon *Plant*	Kontroler		
	PID	*Inverse Neural Model Plant* Simulator ABS	Jaringan Saraf Tiruan Berbasis *Feedforward* yang Mengoptimalkan PID
Time Constant (detik)	0,06	0,36	0,1
Rise Time (detik)	0,09	0,48	0,3
Settling Time (±0,5%) (detik)	12,35	4,7	1,25
Overshoot Maksimum	0 %	150 %	7,65 %
Error Steady-State	0 %	0 %	0 %

Berdasarkan Tabel 4.1 dapat diketahui bahwa kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID mengakibatkan respon *transient plant* yaitu *settling time* ±0,5% menjadi lebih cepat sepuluh kali jika dibandingkan dengan kontroler PID. Kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID juga mampu membuat *plant* memiliki maksimum *overshoot* yang sangat kecil yaitu 7,65 % jika dibandingkan dengan kontroler *inverse neural model plant*, walaupun tidak dapat seperti respon *plant* dengan kontroler PID yang tidak memiliki *overshoot.*

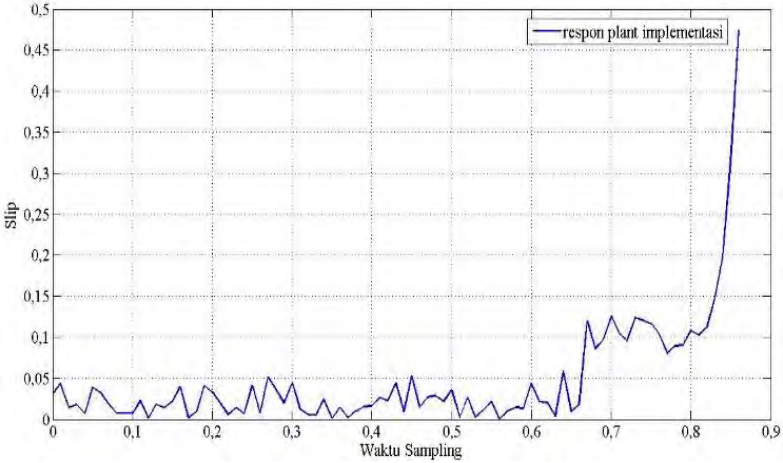
4.3 Implementasi Sistem

Implementasi sistem dilakukan untuk menguji performa kontroler ketika diaplikasikan pada *plant* yang sesungguhnya. *Output* sinyal kontrol berasal dari sinyal *PWM* Arduino. Sinyal kontrol yang dihasilkan oleh kontroler nantinya akan dikalibrasi dan dibatasi dalam nilai 0 – 255. Maka dari itu, perlu diletakan blok *saturation* sebelum blok simulator ABS untuk membatasi sinyal kontrol dari kontroler sehingga Arduino dapat mengolah perintah dari *software* MATLAB. Sinyal PWM dari Arduino nantinya akan masuk ke dalam driver rem mikro elektromagnetik untuk

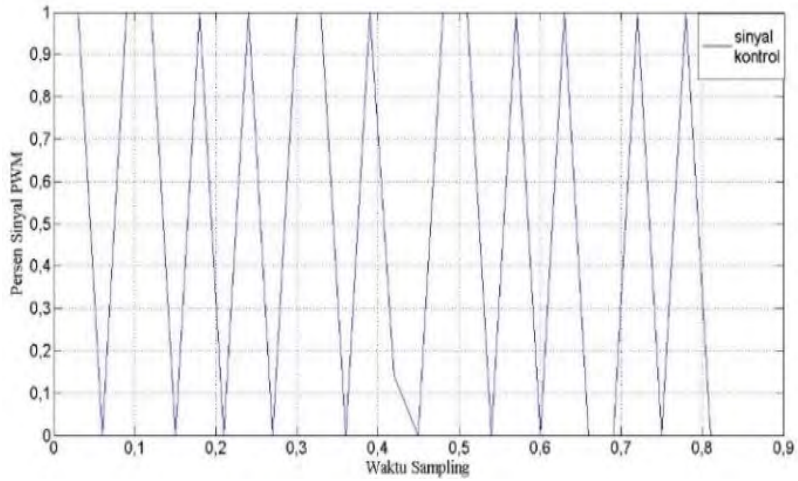
dikuatkan menjadi sinyal PWM dengan rentang tegangan 0 – 12 Volt agar sesuai dengan tegangan kerja rem mikro elektromagnetik.

Implementasi yang dilakukan hanya menggunakan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID. Pengujian kontroler ini dilakukan dengan memberikan *set point* berupa sinyal *step*. Pada simulasi ini, sinyal *step* yang diberikan sebesar 0,2 dengan waktu *sampling* 0,03. Sedangkan untuk penerimaan data sensor *rotary encoder* (blok MATLAB Simulink *serial recieve)* waktu *sampling* dibuat menjadi 0,01. Nilai K_p, K_i, dan K_d yang digunakan sama dengan nilai yang digunakan pada pengujian berupa simulasi. Sedangkan bobot dan arsitektur jaringan saraf tiruan yang digunakan juga sama dengan saat simulasi. Perbedaan nilai waktu *sampling* pada blok peneriman data dan blok kontroler dilakukan agar komunikasi serial antara *software* MATLAB dengan simulator ABS melalui Arduino dapat dilakukan secara *half duplex* sehingga tidak ada data yang saling bertabrakan.

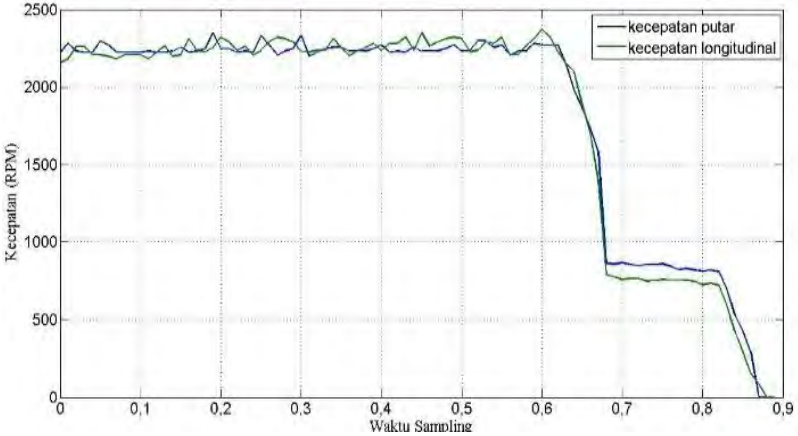
Pada pengujian ini, digunakan dua buah Arduino dimana Arduino yang pertama digunakan untuk mengatur agar motor DC membuat roda atas (kecepatan putar roda mobil) mencapai kecepatan 2193 RPM (sinyal PWM 80 % pada motor DC) sebelum melakukan proses pengereman. Sedangkan Arduino yang kedua digunakan untuk menerima data pembacaan sensor *rotary encoder* dan mengirimkan sinyal kontrol yang telah diolah *software* MATLAB kepada aktuator simulator ABS (rem mikro elektromagnetik).



Gambar 4.7 Respon *Plant* pada saat Implementasi



Gambar 4.8 Sinyal Kontrol Jaringan Saraf Tiruan Berbasis *Feedforward* yang Mengoptimalkan PID saat Implementasi



Gambar 4.9 Penurunan Kecepatan Simulator ABS dengan Kontroler Jaringan Saraf Tiruan Berbasis *Feedforward* yang Mengoptimalkan PID saat Implementasi

Berdasarkan Gambar 4.7 dapat dilihat bahawa kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID mampu membuat sistem ABS bekerja agar slip tidak bernilai 1 (roda mobil

terkunci) pada saat pengereman walaupun belum dapat meregulator besar slip yang dihasilkan menjadi 0,2. Respon *plant* pada saat implementasi (slip yang dihasilkan antara kecepatan *longitudinal* dengan kecepatan putar) berosilasi pada nilai 0,14 sebelum berhenti.

Berdasarkan Gambar 4.9, penurunan kecepatan *longitudinal* mobil dan kecepatan putar roda pada saat pengereman terlihat lebih landai jika dibandingkan dengan penurunan kecepatan pada saat kontroler tidak bekerja (sinyal PWM rem sebesar 100 %). Berdasarkan Gambar 4.7, Gambar 4.8, dan Gambar 4.9, waktu *sampling* yang tertera tidak sesuai dengan waktu asli dikarenakan waktu *sampling* blok *scope* pada *software* MATLAB Simulink mengikuti waktu *sampling* blok pengiriman data (*serial send)*. Masalah ini dapat diatasi dengan membuat waktu *sampling* blok pengiriman data menjadi 1, tetapi perubahan waktu *sampling* ini menyebabkan banyak sinyal kontrol dari kontroler yang telah dirancang terpotong dan hilang.

(*halaman ini sengaja dikosongkan*)

DAFTAR PUSTAKA

[1] _____, "*Cadangan Minyak Dunia Habis Lebih Cepat*", <URL: http://www.hijauku.com/2012/10/03/minyak-habis-lebih-cepat-dari-perkiraan.>, Oktober, 2012.

[2] Bosch, R., *Bosch Electronic Automotive Handbook 1st Edition*, Cambridge: Bentley Publishers, 2012.

[3] John, S., "Artificial Intelligent-Based Feedforward Optimized PID Wheel Slip Controller", in *IEEE African Conf. (AFRICON) 2013*, pp 1-6, 2013.

[4] Chen, C. K., dan Wang, Y. C., "Fuzzy Control for the Anti-lock Brake System". *Soft Computing in Intelligent Systems and Information Processing of the 1996 Asian Fuzzy Systems Symposium,* pp 67-72, 1996.

[5] Sharkawy, A. B., "Genetic Fuzzy Self-Tuning PID Controllers for Antilock Braking System", *Engineering Applications of Artificial Intelligence Journal*, 23(7), pp. 1041–1052, 2010.

[6] Ille, O., *ABS System Control Simulator*, Tallinn: Tallinn University of Technology, 2010.

[7] Kogut, K., "Anti-lock Braking System Modelling and Parameters Identification", in *19th Int. Conf. on Methods and Models in Automation and Robotics (MMAR) 2014*, pp. 342–346, 2014.

[9] _____, *Electromagnetic-actuated Micro Clutches and Brake,* Kawasaki: Miki Pulley Co., Ltd, 2015

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

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

[11] Wheat, D., *Arduino Internals*, New York: Appress, 2011.

[12] Astrom, K. J., *Adaptive Control Second Edition*, New York: Dover Publications, Inc, 2008.

[13] Ljung, L., *System Identification : Theory for The User Second Edition*, New Jersey: Prentice-Hall PTR, 1999.

[14] Ogata, K., *Modern Control Engineering Fifth Edition*, New Jersey: Prentice-Hall PTR, 2010.

[15] Astrom, K. J., *Control System Design,* Santa Barbra: University of California, 2002.

[16] Martiana, E., *Kecerdasan Buatan,* Surabaya: Politeknik Elektronika Negeri Surabaya, 2005.
[17] Nørgaard, M., *Neural Network Based Control System Design Toolkit*, Denmark: Technical University of Denmark, 1996.
[18] Žilková, J., Timko, J., dan Girovský, P., "Nonlinear System Control Using Neural Networks*"*, *Acta Polytechnica Hungarica Journal of Applied Sciences*, 3(4), pp. 85-87, 2006.
[19] Masden, P. P., "Neural Network For Optimization of Existing Control Systems"*,* in *IEEE Int. Conf. on Neural Networks 1995*, 3, pp. 1496-1501, 1995.

BAB 5
PENUTUP

Pada bab ini akan diambil beberapa kesimpulan mengenai hasil pengujian kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID baik berupa simulasi dan implementasi dalam meregulasi slip menjadi 0,2. Pada bab ini juga akan diberikan beberapa saran bagi pembaca jika hendak melanjutkan topik Tugas Akhir tentang *Anti-lock Braking System* (ABS) atau jaringan saraf tiruan.

5.1 Kesimpulan

Berdasarkan hasil pengujian berupa simulasi dan implementasi yang telah dilakukan pada Tugas Akhir ini, maka dapat diperoleh beberapa kesimpulan yaitu:

- Kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID mampu mempercepat respon *transient plant* jika dibandingkan dengan kontroler PID saja, karena jaringan saraf tiruan yang digunakan dilatih untuk menjadi *inverse* dari *plant*.
- Berdasarkan simulasi, *settling time* ±0,5% *plant* dengan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID menjadi lebih cepat sepuluh kali jika dibandingkan dengan kontroler PID.
- Berdasarkan implementasi, kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID mampu membuat sistem ABS bekerja agar slip tidak bernilai 1 pada saat pengereman walaupun belum dapat meregulator besar slip yang dihasilkan menjadi 0,2.
- Jaringan saraf tiruan yang digunakan pada Tugas Akhir ini dilatih dengan metode *backpropagation*, iterasi sebanyak 100.000, laju pembelajaran sebesar 1, dan 8 buah neuron pada lapisan tersembunyi agar dapat menghasilkan *minimum square error* sebesar 0,0028.

5.2 Saran

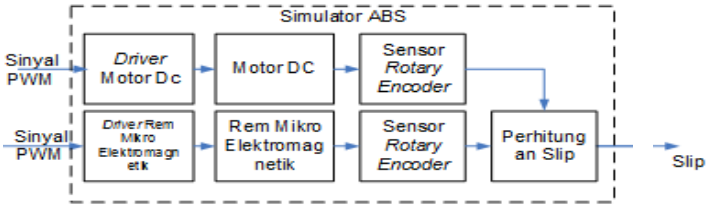
Berdasarkan pengujian simulator ABS yang menggunakan kontroler jaringan saraf tiruan berbasis *feedforward* yang mengoptimalkan PID, maka penulis mencoba untuk memberikan beberapa saran apabila ada

pembaca yang hendak melanjutkan topik Tugas Akhir tentang *Anti-lock Braking System* (ABS) atau jaringan saraf tiruan yaitu:

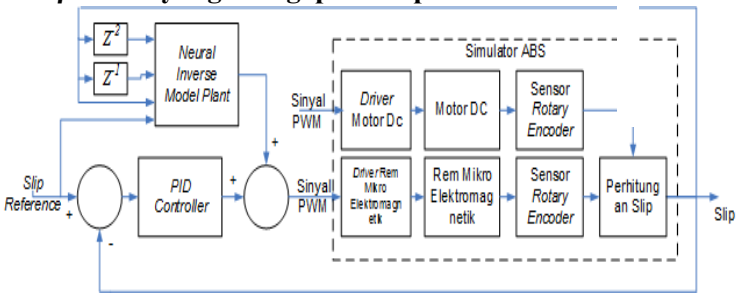
- Jika ingin melakukan implementasi dengan menggunakan *software* MATLAB Simulink dan mikrokontroler Arduino sebagai alat akusisi data, maka sebaiknya memilih waktu *sampling* yang tepat agar sinyal kontrol dari kontroler hasil rancangan tidak hilang dan sesuai dengan *plant* yang ingin dikontrol.
- Algoritma pelatihan *offline invers neural model plant* dapat menggunakan algoritma Levenberg-Marquardt untuk melihat apakah berpengaruh terhadap kecepatan jaringan saraf tiruan untuk mencapai nilai lokal minimum *error* terhadap target pembelajaran yang telah diberikan.

LAMPIRAN 1
SKEMA ALAT

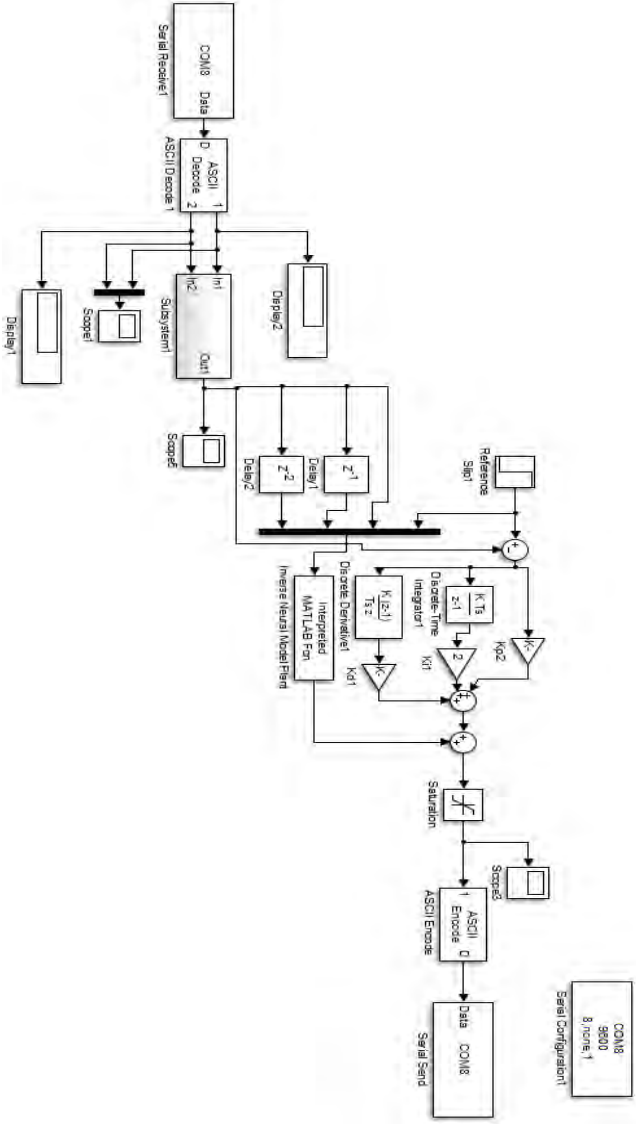
A. Diagram Blok Simulator ABS



B. Digaram Blok Kontroler Jaringan Saraf Tiruan berbasis *Feedforward* yang Mengoptimalkpan PID



C. Blok Diagram Implementasi Kontroler Jaringan Saraf Tiruan Berbasis *Feedforward* yang Mengoptimalkan PID



LAMPIRAN 2
PROGRAM YANG DIGUNAKAN

A. Program *Pembacaan* Sensor *Rotary Encoder*

```
int encoder1  = 4 ;
 int encoder2  = 3 ;

void setup() {
   Serial.begin(9600);
   pinMode(encoder1,INPUT);  //longitudinal
   pinMode(encoder2,INPUT);  //putar
   pinMode(13, OUTPUT);
}

void loop() {
  kec_longitudinal();
  kec_putar();
}

void kec_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(){
  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);
}
```

B. Program Sinyal PWM untuk Motor DC dan Rem Mikro Elektromagnetik

```
int pinrem= 4 ;
int rem= 9 ;

void setup() {
  pinMode(pinrem,INPUT)
  pinMode(rem,OUTPUT);
}

void loop() {
  if ( digitalRead(pinrem) == HIGH) {
    digitalWrite(rem,HIGH);
    delayMicroseconds(1998);
    digitalWrite(rem,LOW);
    delayMicroseconds(2);
  }
  else {
    digitalWrite(motor,HIGH);
    delayMicroseconds(1000);
    digitalWrite(motor,LOW);
    delayMicroseconds(1000);
  }
}
```

C. Program Kontroler Jaringan Saraf Tiruan Berbasis *Feedforward* Tahap Pelatihan *Offline* dengan *Software* MATLAB

```
clear all
clc
```

```
load y.mat
load yk.mat
load yk1.mat
load yk2.mat
load u.mat
ninl=4;
nhl=8;
nol=1;
alph=1;
alpo=1;
sse=1;
wih=-0.5+1*rand(nhl,ninl);
woh=-0.5+1*rand(nhl,nol);
vih=-0.5+1*rand(nhl,1);
voh=-0.5+1*rand(nol,1);
erk=zeros(134,1);
xx=[y yk yk1 yk2];

for epoch=1:100000
   if sse>0.0000001
      for d=1:134
         xin=xx(d,:);
         yout=u(d);
         for j=1:nhl
            for l=1:ninl
               z_h(l)=wih(j,l)*xin(l);
               zh(j)=sum(z_h)+vih(j);
               yh(j)=1/(1+exp(-zh(j)));
            end
         end
         for k =1:nhl
            z_o(k)=woh(k)*yh(k);
         end
         zo=sum(z_o)+voh;
         yo=1/(1+exp(-zo));
         er=yout-yo;
         erk(d)=er*er;
         delo=er*yo*(1-yo);
         voh=voh+alpo*delo;
```

```
            for m=1:nhl
               erh(m)=delo*woh(m);
               delh(m)=erh(m)*yh(m)*(1-yh(m));
               woh(m)=woh(m)+delo*alpo*yh(m);
               vih(m)=vih(m)+alph*delh(m);
               for n=1:ninl
                  wih(m,n)=wih(m,n)+delh(m)*alph*xin(n);
               end
            end
         end
         iterasi=epoch
         sse=sum(erk)/134
      end
   end
```

D. Program Kontroler Jaringan Saraf Tiruan Berbasis *Feedforward* Tahap *Online* dengan *Software* MATLAB

```
function yo = berhasil(xin)
global in1 in2 in3 in4 ninl nhl nol alph alpo wih woh vih voh xx z_h
zh yh z_o zo


in1 = xin(1);
in2 = xin(2);
in3 = xin(3);
in4 = xin(4);

ninl=4;
nhl=8;
nol=1;
alph=1;
alpo=1;
wih=[-0.1664993 -28.5693 -3.94958 14.381079;-2.941725 -
99.4261 31.7115 128.0843;-0.83012 1.708512 -0.6503 -1.9913
; 4.6606 -158.3008 -37.40818 56.1815;-6.282341 108.675
38.2642 -11.33061; -2.5023 91.37408 14.1425 -40.5939 ; -0.89
68.56507 2.110954 -45.2719 ;0.7066 -47.46425 -6.4794
24.3257];
woh=[7.145004 ; 12.6739271 ; -1.5508 ; 40.4834 ; -33.3875 ; -
17.52207 ; -13.945 ; 9.682];
```

```
vih=[1.17652 ; -12.847939 ; -3.2692606 ; 24.1463272 ; -28.8842 ; -
14.53841 ; -6.7403885 ; 3.52559];
voh=[-2.473539295830077];
xx=[in1 in2 in3 in4];

for j=1:nhl
   for l=1:ninl
      z_h(l)=wih(j,l)*xx(l);
      zh(j)=sum(z_h)+vih(j);
      yh(j)=1/(1+exp(-zh(j)));
   end
end
for k =1:nhl
   z_o(k)=woh(k)*yh(k);
end
zo=sum(z_o)+voh;
yo=1/(1+exp(-zo));
```

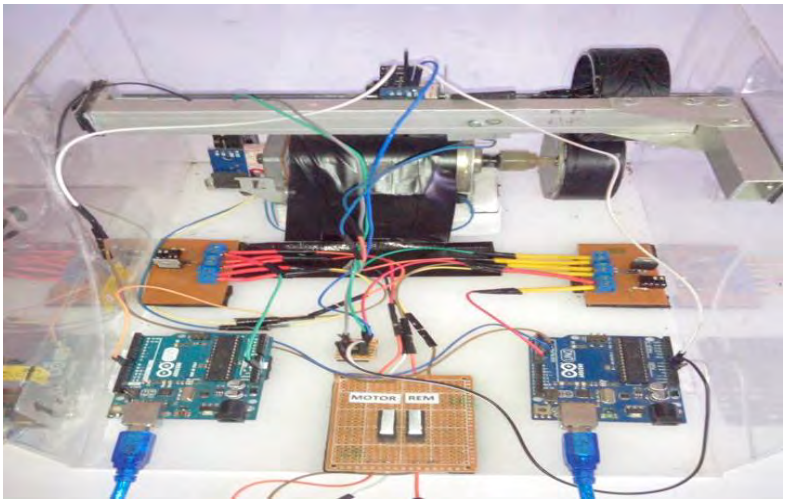
E. Program Arduino untuk Pengujian Berupa Implementasi yang Menggunakan Kontroler Jaringan Saraf Tiruan Berbasis *Feedforward* yang Mengoptimalkan PID

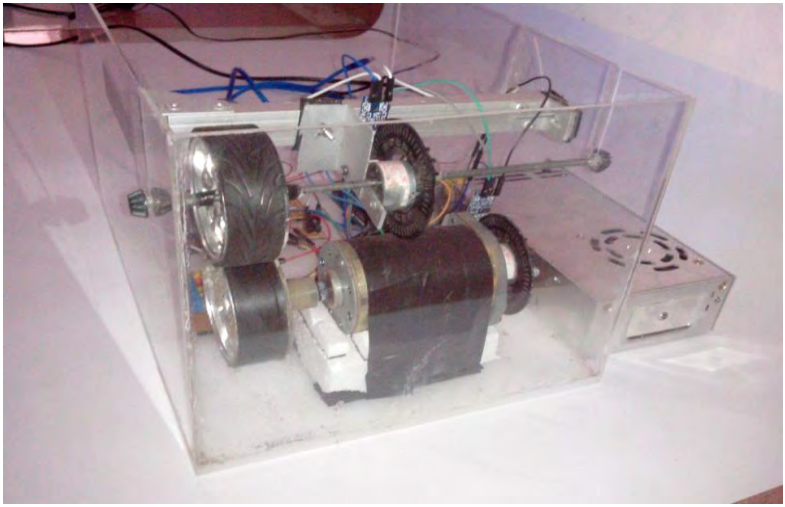
```
#include <Wire.h>
int encoder1 = 7;
int encoder2= 5;
float percent ;
float b;
char tampilan[1];
char tampilan1[1];
char tampilan2[1];

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

void loop(){
  if (Serial.available()>0) {
```

```
  percent=Serial.parseFloat();
  b = percent*255;
  analogWrite(9,b);}
  bacaEncoderLong();
  bacaEncoderAng();
}

void bacaEncoderLong(){
  unsigned long waktu = 0;
  float jeda = 0;
  float kecepatan_longitudinal = 0;
  word vel = 0;
  waktu = pulseIn(encoder1, LOW, 6884);
  jeda = waktu;
  jeda = jeda / 1000000;
  kecepatan_longitudinal = (1 / (jeda * 110 ))* 60 ;
  vel =  kecepatan_longitudinal + 150;
  sprintf(tampilan,"*%4d",vel);
  Serial.print(tampilan);
}

void bacaEncoderAng(){
  unsigned long waktu = 0;
  float jeda = 0;
  float kecepatan_putar = 0;
  word rpm = 0;
  waktu = pulseIn(encoder2, LOW, 6884);
  jeda = waktu;
  jeda = jeda / 1000000;
  kecepatan_putar = (1 / (jeda * 110)) * 60 ;
  rpm =  kecepatan_putar;
  sprintf(tampilan1,"#%4d",rpm);
  Serial.print(tampilan1);
}
```

LAMPIRAN 3
FOTO ALAT

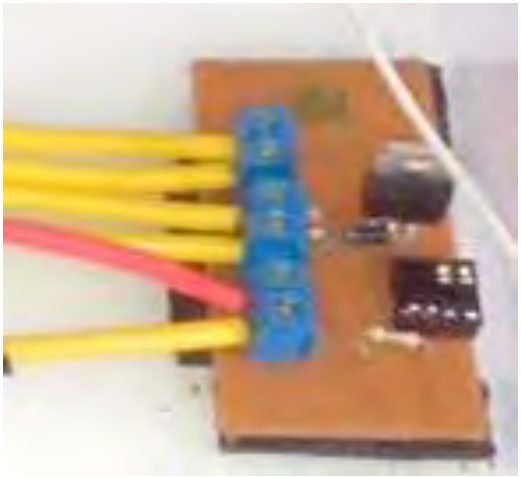
a. Simulator ABS



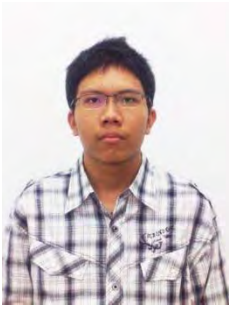


b. Bentuk Fisik Driver Motor DC

d. Bentuk Fisik Driver Rem Elektromagnetik



RIWAYAT HIDUP



Benny Adijaya Joesoep lahir di Semarang pada tanggal 07 Oktober 1993. Penulis merupakan anak kedua dari pasangan (alm.) Harsono Joesoep dan Theresia Melani Yadin. Penulis memulai pendidikan formal di SD Kebon Dalem Semarang, SMP Kebon Dalem Semarang, dan SMA Kolese Loyola Semarang. Setelah menamatkan pendidikan sekolah menengah atas, penulis meneruskan ke jenjang perguruan tinggi Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember pada bidang studi Teknik Sistem Pengaturan. Penulis dapat dihubungi melalui email : **benny.adijaya@gmail.com**