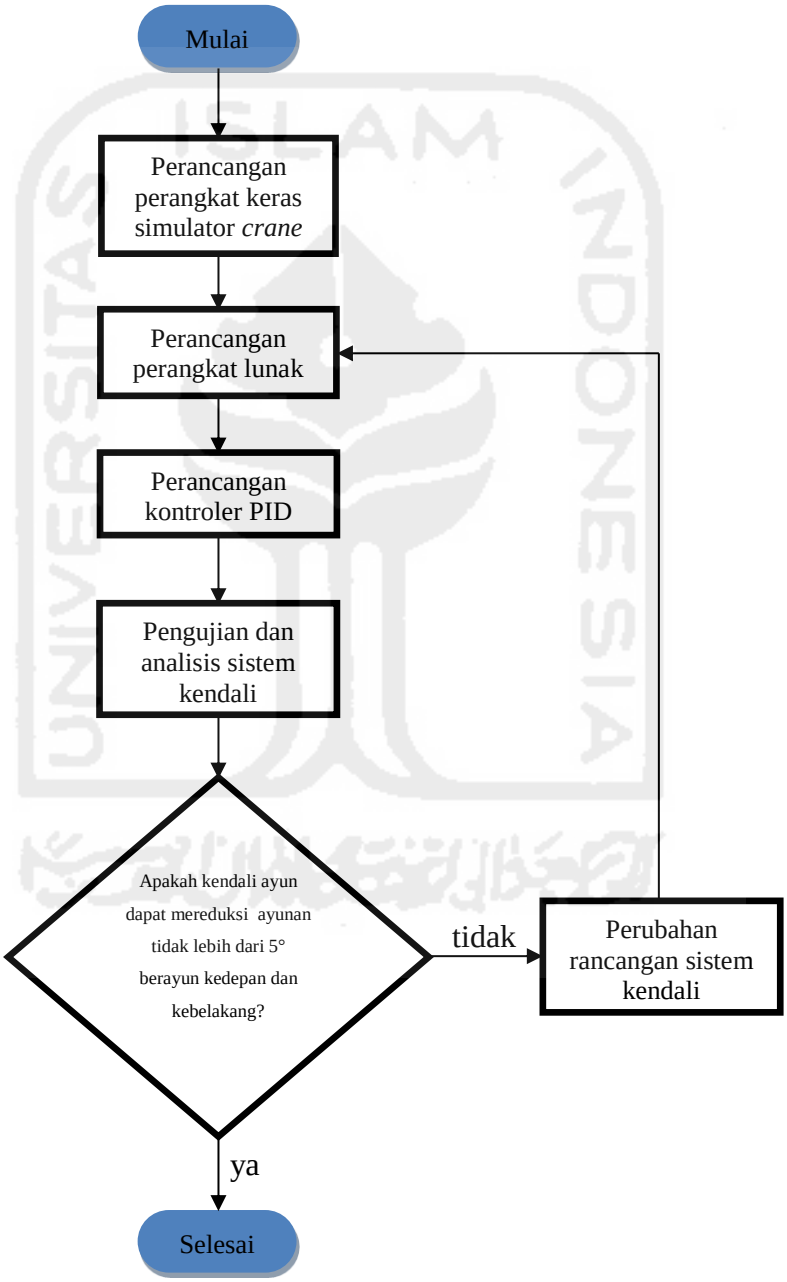
BAB 3
METODOLOGI PENELITIAN

3.1 Alur Penelitian

Agar penelitian ini lebih mudah untuk dilakukan maka dibuatlah alur penelitian yang terdari dari beberapa tahap seperti Gambar 3-1 dibawah ini.

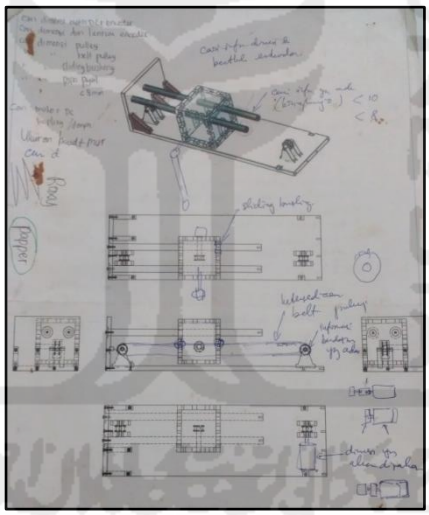


Gambar 3 - 1 Alur Penelitian

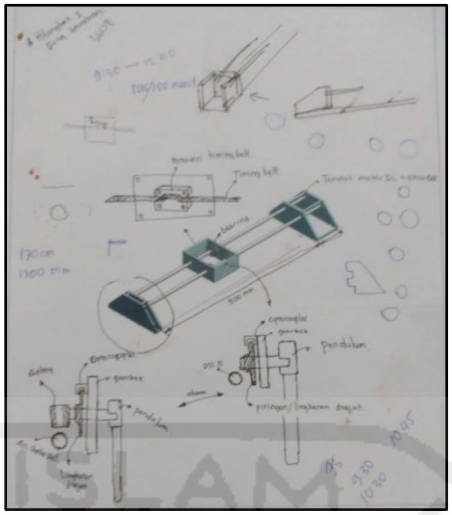
3.2 Perancangan Simulator *Crane*

3.2.1 Perancangan Perangkat Keras Mekanik

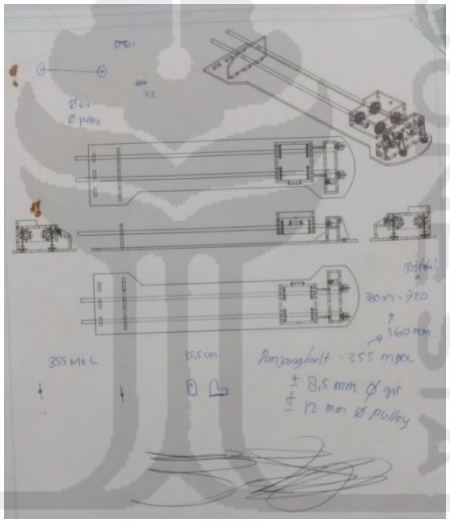
Perancangan dilakukan dengan menggunakan perangkat lunak Solidworks 2012, dimana Solidworks dapat menunjukan visualisasi berupa bentuk 2 dimensi ataupun 3 dimensi alat yang dirancang. Proses perancangan dilakukan dengan beberapa tahap yaitu gambaran awal untuk mengetahui bagaimana bentuk dari simulator, mekanisme gerak simulator, dimana letak motor DC, letak sensor pembaca gerakan batang ayun, letak Arduino ditempatkan, panjang lintasan simulator, dimensi simulator, bagaimana menggerakan troli yang dihubungkan dengan *belt*, penggerak troli apakah menggunakan *bushing* atau *sliding bushing*, dan lain sebagainya. Hasil gambaran awal perancangan dapat dilihat pada Gambar 3-2 hingga 3-4.



Gambar 3 - 2 Gambaran Awal Peracangan Pertama

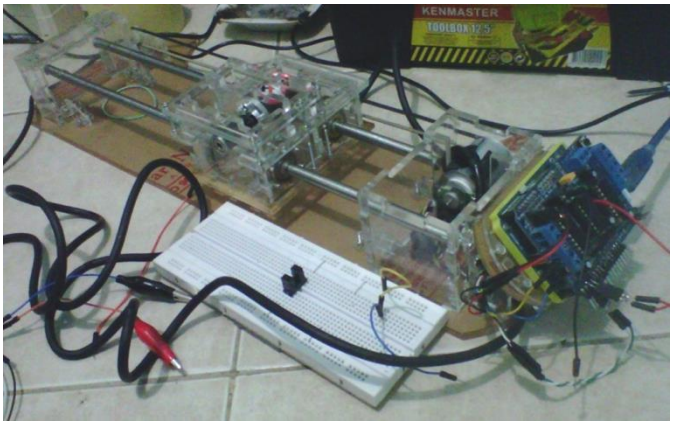


Gambar 3 - 3 Gambaran Awal Perancangan Kedua



Gambar 3 - 4 Gambaran Awal Perancangan Ketiga

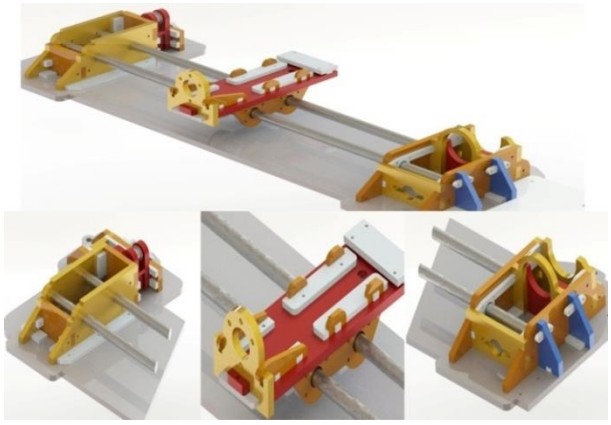
Pada gambaran awal perancangan ketiga akan dilakukan pembuatan alat simulator menjadi sebuah produk untuk mengetahui bentuk nyata dari perancangan yang telah dilakukan serta agar dapat mengetahui kekurangan seperti apa yang akan dapat diubah menjadi lebih baik. Gambar 3-5 berikut ini menunjukan bentuk nyata dari perancangan alat.



Gambar 3 - 5 Bentuk Nyata Simulator Crane Berdasarkan Perancangan Yang Telah Dilakukan.

Pada perancangan yang sudah terealisasi menjadi alat simulator *crane*, masih terdapat kendala diantaranya penggunaan *bearing* pada troli masih menimbulkan gesekan antara *bearing* dengan besi pejal sebagai lintasan troli yang mengakibatkan gerakan troli akan menjadi berat, lebar *belt* yang digunakan masih terlalu kecil yang mengkibatkan saat poros motor DC berputar *belt* tersebut mudah lepas, tempat meletakan perangkat Arduino yang masih kurang tersusun dan terkesan terlalu penuh, dan bentuk troli yang terlalu besar sehingga akan membuat putaran poros motor DC untuk menggerakan troli akan semakin berat.

Berdasarkan kendala yang didapat, maka dilakukan perubahan rancangan seperti bentuk troli dibuat lebih ramping, penggerak troli sebelumnya menggunakan *bearing* kemudian diganti menjadi *sliding bushing* karena sesuai dengan arah gerakan troli yang bergerak secara linier, dan bentuk dari simulator *crane* dibuat sangat simpel guna memaksimalkan kinerja simulator *crane*. Perubahan rancangan alat simulator *crane* dapat dilihat pada Gambar 3-6.



Gambar 3 - 6 Hasil Akhir Perancangan Simulator Crane Yang Dipilih.

Pada Gambar 3-7 adalah hasil pembuatan produk nyata berdasarkan hasil perancangan simulator *crane* yang telah dilakukan.

Gambar 3 - 7 Bentuk Nyata Simulator Crane Berdasarkan Hasil Perancangan Terakhir.

3.2.2 Peralatan dan Bahan

Pada penelitian ini ada beberapa perangkat yang digunakan yaitu perangkat keras elektronik dan perangkat lunak. Perangkat keras elektronik meliputi motor DC enkoder, *incremental rotary encoder*, Arduino Mega, motor *shield*, kabel usb, penaik tegangan, adaptor, dan PC (*Personal Computer*). Perangkat lunak yang digunakan meliputi OS (*Operating system*) PC, Matlab Simulink, dan Arduino IDE.

3.2.2.1 Perangkat Keras Elektronik

A. Motor DC Enkoder

Dalam penelitian ini menggunakan motor DC Mabuchi RS-385PH dengan tipe enkoder inkremental 448 AB *phase*. Bentuk perangkat keras dari motor DC dapat dilihat pada Gambar 3-8.

Gambar 3 - 8 Motor DC Mabuchi RS-385PH dengan enkoder 448 AB phase.

Motor ini termasuk jenis motor DC bersikat. Pada motor DC bersikat rotornya bersifat elektromagnetik dan bagian statornya bersifat magnet alami. Spesifikasi motor DC dapat dilihat pada Tabel 3-1.

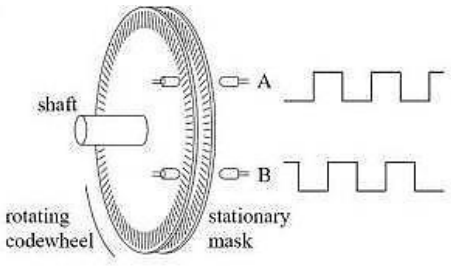
Tabel 3 - 1 Spesifikasi Motor DC Mabuchi RS-385PH dengan enkoder 448 AB phase.

Resolusi enkoder	448 P/R
Tegangan input enkoder	5V
Fase enkoder	A dan B
Tegangan supply motor	12 VDC – 30 VDC
Arus motor tanpa beban	70 mA
Kecepatan motor tanpa beban	8.700 Rpm
Torsi motor	84,6 g-cm (8,3 mN-m)
Panjang bodi motor	37,8 mm
Diameter bodi motor	29,2 mm (dlm) / 30 mm (luar)
Panjang poros motor	75,2 mm
Diameter poros motor	2,3 mm
Berat	91 g

Enkoder inkremental terdiri dari dua *track* atau *single track* dan dua sensor yang disebut *channel* A dan B (Gambar 3-9). Ketika poros berputar, deretan pulsa akan muncul dari masing-masing *channel* pada frekuensi yang proporsional dengan kecepatan putar. Hubungan antara fasa *channel* A dan B menghasilkan arah putaran. Dengan menghitung jumlah pulsa yang terjadi pada resolusi piringan maka putaran dapat diukur. Sedangkan untuk mengetahui arah putaran diketahui dari *channel* mana yang *leading* terhadap *channel* satunya.

Arah putaran yang terjadi antara kedua *channel* tersebut akan selalu berbeda fasa seperempat putaran (*quadrature signal*). Motor DC yang digunakan dalam penelitian ini menggunakan *quadrature* maka resolusi enkoder dikalikan 4 setiap satu kali putaran, sebab untuk enkoder *quadrature* mempunyai 2 *channel* seperti yang sudah dijelaskan diatas. Setiap *channel*-nya memilik 2 kondisi yaitu *high* dan *low*. Jika enkoder mempunya 2 *channel* berarti enkoder juga mempunyai 4 kondisi. Resolusi yang dimiliki oleh enkoder keluarannya dikalikan 4 untuk mendapat nilai pulsa yang *valid*. Seringkali terdapat *output channel*

ketiga, disebut *index*, yang menghasilkan satu pulsa per putaran berguna untuk menghitung jumlah putaran yang terjadi.



Gambar 3 - 9 Susunan piringan enkoder inkremental (Angin, 2009).

B. *Incremental Rotary Encoder*

Pada penelitian ini digunakan *incremental rotary encoder* merk Autonics dengan tipe E30S4-500-3-T-24. Bentuk dari perangkat tersebut ditunjukan pada Gambar 3-10.

Gambar 3 - 10 Autonics Incremental Rotary Encoder.

Tabel 3 - 2 Spesifikasi Incremental Rotary encoder.

Seri	E30S4-500-3-T-24
Resolusi Enkoder	500 P/R
Power Supply	12-24 VDC ± 5%
Fase Enkoder	A, B, dan Z
Diameter bodi enkoder	30 mm
Panjang bodi enkoder	31,5 mm
Panjang poros Enkoder	11,5 mm
Diameter poros Enkoder	4 mm
Berat	80 g

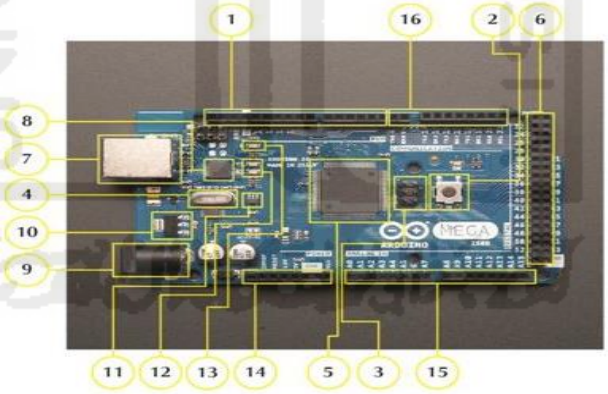
Spesifikasi lebih lengkap mengenai perangkat keras *rotary encoder* dapat dilihat pada halaman lampiran tugas akhir ini.

C. Arduino Mega

Mikrokontroler yang digunakan adalah Arduino Mega 2560 R3 yang berbasis Atmega 2560. Memiliki 54 pin *input*/*output* digital, 14 pin diantaranya digunakan sebagai pin pwm, 16 pin analog, 4 UART (*port serial*), 16 Mhz osilator kristal, koneksi usb, *jackpower*, ICSP *header*, dan tombol reset. Untuk mendukung mikrokontroler agar dapat digunakan, cukup hanya menghubungkan *board* Arduino ke komputer dengan menggunakan kabel usb atau baterai untuk menjalankannya. Gambar 3-11 menunjukan bentuk dari Arduino Mega 2560 R3.

Gambar 3 - 11 Arduino Mega 2560 R3 (Faekar, 2015).

Komponen-komponen dari Arduino Mega 2560 R3 dapat dilihat pada Gambar 3-12 dibawah ini.



Gambar 3 - 12 Komponen Arduino Mega 2560 R3 (Faekar, 2015).

Keterangan komponen-komponen Arduino Mega 2560 R3 diatas :

1. 14 pin *input*/*output* digital (0-13)

   Berfungsi sebagai *input* atau *output*, dapat diatur oleh program. Khusus untuk 6 buah pin 3, 5, 6, 9, 10 dan 11, dapat juga berfungsi sebagai pin analog atau *output* pin PWM dimana tegangan *output*-nya dapat diatur.

Nilai sebuah pin *output* analog dapat diprogram 0-255. Dimana hal itu mewakili nilai tegangan 0-5V.

2. Tombol reset S1

   Untuk me-*reset* papan sehingga program akan dimulai kembali dari awal. Perhatikan bahwa tombol reset ini bukan untuk menghapus program atau mengosongkan mikrokontroler.

3. ICSP (*In-Circuit Serial Programing*)

   Port ICSP memungkinkan pengguna untuk memprogram mikrokontroler secara langsung, tanpa melalui *bootloader*. Umumnya pengguna Arduino tidak melakukan ini sehingga ICSP tidak terlalu dipakai walaupun disediakan.

4. Q1-Kristal (*quartz crystal oscillator*)

   Jika mikrokontroler dianggap sebagai sebuah otak, maka kristal adalah jantung-nya karena komponen ini menghasilkan detak-detak yang dikirim kepada mikrokontroler agar melakukan sebuah operasi untuk setiap detak-detaknya. Kristal ini dipilih yang berdetak 16 juta kali per detik (16Mhz).

5. IC 1-Mikrokontroler Atmega

   Mikrokontroler Atmega merupakan komponen utama dari papan Arduino, di dalamnya terdapat CPU, ROM, dan RAM.

6. Pin digital *extended* (22-53)

   Pada Arduino Mega disediakan pin digital tambahan sejumlah 31 pin. Apabila pin digital 0-13 sudah dipakai makan dapat menggunakan pin ini sebagai *input*/*output*-nya. Namun perlu diingat jika pin-pin tersebut tidak dapat digunakan sebagai pin PWM.

7. USB

   USB berfungsi sebagai pemuat program dari komputer ke dalam mikrokontroler Arduino, sebagai komunikasi serial antara mikrokontroler dan sebagai pemberi masukan daya listrik kepada mikrokontroler.

8. FTDI *Chip*

   Chip FTDI pada Arduino Mega 2560 R3 adalah chip Atmega 16U2 yang berfungsi sebagai konverter USB ke serial.

9. Sumber daya eksternal

   Mikrokontroler Arduino jika hendak disuplai dengan sumber daya eksternal dapat diberikan tegangan DC antara 7-12V.

10. Regulator 5 Volt
11. Regulator 3,3 Volt
12. LED TX/RX

    LED ini merupakan indikator ketika Arduino mengirim (*transmitting*)/menerima (*receiving*) data. Contoh pada saat sebuah program diunggah ke Arduino maka LED ini akan menyala berkedip.

13. LED Arduino

    LED *built-in* yang dapat dikendalikan melalui pin digital 13.

14. Pin catu daya

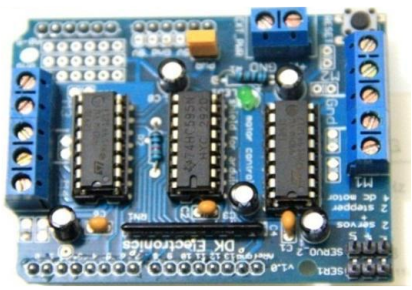
    Pada barisan pin ini merupakan catu daya yang diperlukan dalam penggunaan sensor ataupun aktuator.

15. 16 pin *input* analog (0-15)

    Pin ini sangat berguna untuk membaca tegangan yang dihasilkan oleh sensor analog, seperti sensor suhu. Program dapat membaca nilai sebuah pin *input* antara 0-1023, dimana hal itu mewakili nilai tegangan 0-5 V.

D. Motor *Shield*

Motor *shield* yang digunakan pada penelitian ini adalah Adafruit v1 yang memungkinkan Arduino untuk mengontrol 4 motor DC, 2 motor *stepper*, dan 2 motor *servo*. Pin-pin untuk menghubungkan kabel motor DC yaitu pada pin M1, M2, M3, dan M4. Motor *shield* ini menggunakan IC L293D yang pengiriman arus keluaran sampai dengan 1,2 A untuk setiap salurannya. Bentuk perangkat tersebut dapat dilihat pada Gambar 3-13 dibawah ini.



Gambar 3 - 13 Motor shield Adafruit v1.

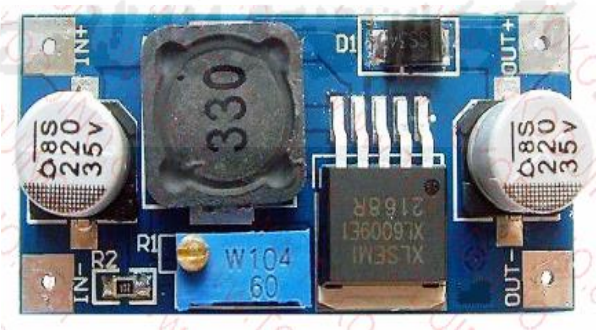
E. Kabel USB

Kabel USB adalah jenis kabel standar yang digunakan untuk menghubungkan atau mengkomunikasikan antara PC dengan perangkat Mikrokontroler. Bentuk perangkat tersebut dapat dilihat pada Gambar 3-14.

Gambar 3 - 14 Kabel USB.

F. DC to DC *Boster Converter*

DC to DC *booster converter* adalah module yang berfungsi untuk mengubah tegangan masukan (*input*) menjadi tegangan keluaran (*output*) yang lebih tinggi. Tegangan *output* dapat diatur dengan cara memutar trimpot. Jika terjadi perubahan tagangan *input*, maka tegangan *output* akan tetap stabil karena module ini juga berfungsi sebagai regulator. Regulator adalah rangkaian atau pengatur tegangan keluaran dari sebuah catu daya agar efek dari naik turunnya tegangan jala-jala tidak mempengaruhi tegangan catu daya sehingga menjadi stabil. Bentuk perangkat tersebut ditunjukan pada Gambar 3-15.



Gambar 3 - 15 DC to DC Booster *Converter*.

G. Adaptor

Adaptor yang digunakan pada penelitian ini menggunakan merk NAMICHI dengan tegangan *output* yang dapat diatur sesuai keiinginan pemakai. Variasi tegangan yang disediakan bernilai 1,5V, 3V, 4,5V, 6V, 7,5V, 9V, dan 12V. Ampere yang dimiliki sebesar 1,2A. Bentuk dari perangkat tersebut ditunjukan pada Gambar 3-16 dibawah ini.

Gambar 3 - 16 Adaptor Namichi.

H. PC (*Personal Computer)*

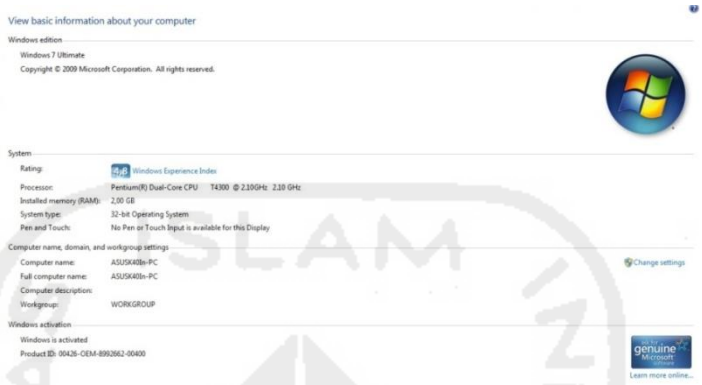
Perangkat komputer yang digunakan dalam penelitian ini adalah merk ASUS K40IN yang menggunakan proccesor Pentium (R) Dual-Core CPU T4300 @2,10Ghz dengan dukungan grafik NVIDIA Geforce G102M Cuda 512Mb dengan RAM 2Gb dan menggunakan sistem operasi 32-bit. Perangkat komputer ditunjukan pada Gambar 3-17.

Gambar 3 - 17 PC (*Personal Computer*).

3.2.2.2 Perangkat Lunak

A. OS (*Operating System) PC*

Pada penelitian ini digunakan OS (*operating system*) keluaran WINDOWS dengan versi Windows 7 *Ultimate* 32 bit. Bentuk perangkat lunak tersebut ditunjukan pada Gambar 3-18.



Gambar 3 - 18 Tampilan OS Windows 7 Ultimate.

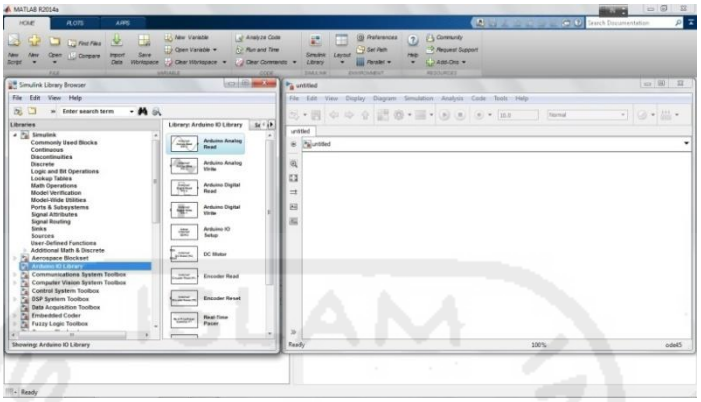
B. Matlab (Matrix Laboratory) R2014a

Secara singkat Matlab adalah suatu perangkat lunak yang dikembangkan oleh pengembang untuk memudahkan penggunanya dalam bidang komputasi, visualisasi, dan pemrograman. Matlab mempunyai *feature* Simulink *library* (Gambar 3-19), dengan *feature* tersebut dapat digunakan untuk pengembangan algoritma pada Arduino.

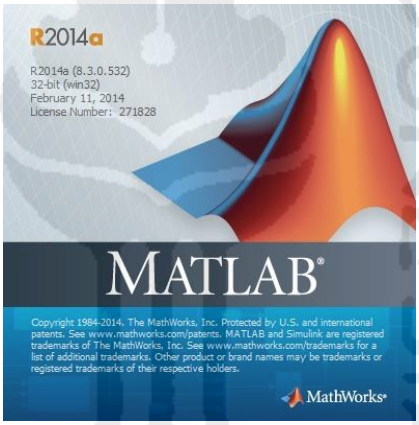
Simulink *library* juga digunakan untuk membuat algoritma pada sistem kontrol dan aplikasi robotika. Selain itu program simulink dapat digunakan untuk mensimulasikan sistem, artinya mengamati dan menganalisa perilaku dari tiruan sistem. Tiruan sistem ini diharapkan mempunyai perilaku yang sangat mirip dengan sistem fisik. Simulink sebagai aplikasi untuk beberapa dunia industri seperti desain model. Simulink yang dirancang untuk platform Arduino meliputi:

- Instalasi otomatis dan konfigurasi.
- *Library* dari blok Simulink yang terhubung ke Arduino I/O seperti analog *input* dan *output*, serial untuk menerima dan mentransmisikan program.
- Parameter interaktif tuning dan pemantauan sinyal aplikasi yang berjalan pada Arduino Mega sedangkan pada Arduino Uno tidak tersedia.

Matlab tersedia untuk 32-bit dan 64-bit *Microsoft Windows*, dan 64-bit *Mac* OS X. Dalam penelitian ini digunakan Matlab Versi R2014a (8.3.0.532) 32-bit (win32). Bentuk dari Matlab itu sendiri dapat dilihat pada Gambar 3-20.



Gambar 3 - 19 Matlab Simulink library.



Gambar 3 - 20 Perangkat lunak Matlab.

C. Arduino IDE

Perangkat lunak Arduino yang digunakan adalah *driver* dan IDE (*Integrated Development Environtment*). IDE Arduino adalah perangkat lunak yang ditulis dengan menggunakan *java*. IDE Arduino terdiri dari :

1. *Editor* program

   Sebuah *window* yang memungkinkan pengguna menulis dan mengedit program dalam bahasa *proccesing*.

2. *Compiler*

   Sebuah modul yang mengubah kode program (bahasa *processing*) menjadi kode yang bisa dipahami oleh mikrokontroler.

3. *Uploader*

   Sebuah modul yang memuat kode biner dari komputer menuju ke dalam *memory* di dalam Arduino.

Tampilan perangkat lunak Arduino IDE dapat dilihat pada Gambar 3-21 dibawah ini, versi yang digunakan adalah versi 1.6.5.

Gambar 3 - 21 Arduino IDE 1.6.5.

3.2.3 Perakitan Perangkat Keras Mekanik dengan Elektronik

Pada perancangan simulator telah ditentukan bahwa yang ditunjukan pada Gambar 3-6 dan 3-7 adalah sebuah perancangan alat yang akan digunakan dalam melakukan penelitian tugas akhir ini. Bagian-bagian simulator tersebut terdiri dari rumahan (*casing*) motor DC, rumahan simulator troli, rumahan penyangga besi lintasan troli dan mekanisme gerak simulator troli akibat putaran pada poros motor DC terhadap *pulley*, dan tempat untuk meletakan perangkat mikrokontroler Arduino.

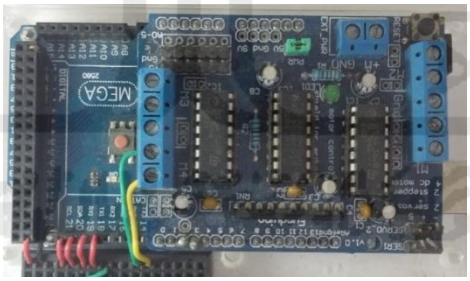
Tahapan perakitan simulator alat tugas akhir ini antara rancangan simulator yang telah direalisasikan dengan perangkat keras yang telah dipilih seperti Arduino, Motor *shield*, *rotary encoder*, Motor DC, *belt*, dan lain sebagainya akan dijelaskan dibawah ini.

Simulator *crane* yang dirancang memiliki dua derajat kebebasan atau dalam bahasa asing sering disebut 2DOF (*two degree of freedom*). Letak DOF pada simulator *crane* yaitu pada lintasan horisontal simulator troli dan pada poros rotasi batang ayun terhadap *incremental rotary encoder* pada simulator troli. *Incremental rotary encoder* digunakan untuk membaca simpangan/ayunan pada batang ayun saat simulator troli bergerak. Perancangan simulator *crane*

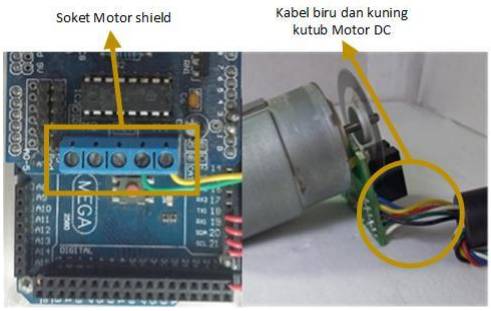
menggunakan mikrokontroler Arduino Mega 2560 R3 sebagai penghubung antara *software* Matlab Simulink dengan simulator *crane*. Motor *shield* digunakan untuk menentukan arah dan kecepatan pada motor DC. Bentuk dari Arduino Mega 2560 R3 dan Motor *shield* adafruit v1 dapat dilihat pada Gambar 3-11 dan 3-13.

Perakitan komponen perangkat keras pada simulator *crane* terlebih dahulu memasangkan motor *shield* ke Arduino. Pemasangan motor *shield* ke Arduino dilakukan dengan cara menghubungkan/menggabungkan soket pada motor *shield* ke soket Arduino. Notasi pada soket motor *shield* harus sesuai dengan notasi soket pada Arduino ketika dihubungkan.

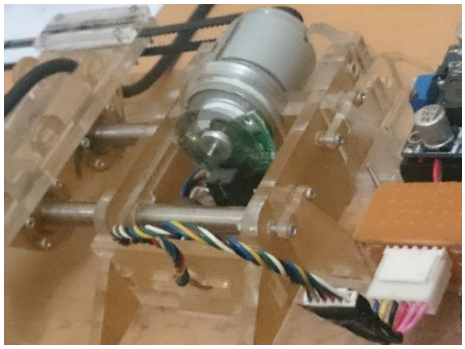
Dalam penelitian ini digunakan motor DC Mabuchi RS-385PH dengan enkoder 448 AB *phase*. Bentuk perangkat tersebut dapat dilihat pada Gambar 3-8. Kutub positif dan negatif motor DC disambungkan ke soket motor *shield* menggunakan kabel jumper sesuai kutub yang ada pada motor *shield*. Kemudian pasangkan motor DC pada rumahan/*casing* yang telah dibuat dengan menggunakan baud M2 untuk mengencangkannya. Lebih jelasnya mengenai pemasangan motor *shield* ke Arduino, menghubungkan kabel *jumper* dari motor DC ke soket motor *shield*, dan memasangkan motor DC ke rumahan/*casing* simulator ditunjukan pada Gambar 3-22 hingga 3-24.



Gambar 3 - 22 Pemasangan Motor *Shield* ke Papan Arduino.

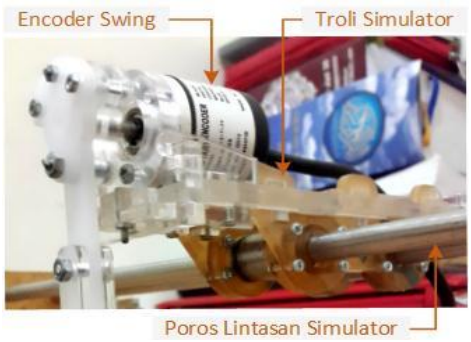


Gambar 3 - 23 Kutub Positif dan Negatif Motor DC Dihubungkan ke Motor *Shield*.

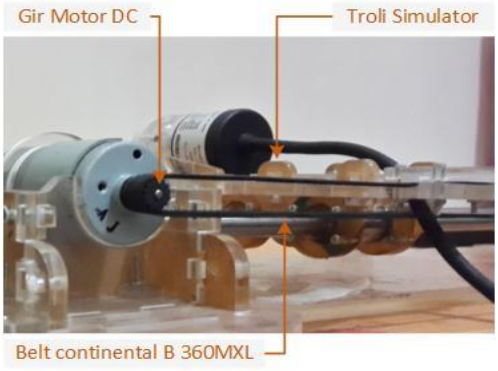


Gambar 3 - 24 Pemasangan Motor DC Pada Rumahan/*Casing* Simulator.

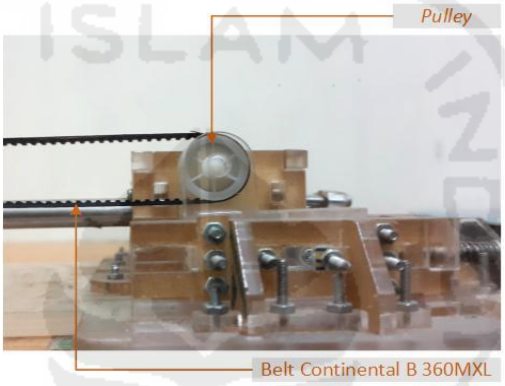
Setelah Arduino, motor *shield*, dan motor DC dipasangkan pada rumahan/*casing*, langkah berikutnya adalah memasang perangkat *incremental rotary encoder* beserta troli pada simulator. *Incremental rotary encoder* pada simulator troli selanjutnya disebut dengan *encoder* posisi ayun. *Encoder* posisi ayun berfungsi untuk membaca gerakan ayunan pada batang ayun yang dihasilkan oleh jalannya motor DC. Troli dihubungkan ke gir yang terdapat pada poros motor DC dengan menggunakan *belt* Continental B 360MXL terhadap *pulley* yang terdapat pada ujung simulator agar dapat menggerakan troli. Setelah itu batang ayun dihubungkan/dipasangkan pada poros *encoder* posisi ayun. *Encoder* posisi ayun memiliki 2000 pulsa dalam satu kali rotasi berdasarkan eksperimen yang telah dilakukan. Pada saat troli bergerak, ayunan yang terjadi akan dibaca oleh *encoder* posisi ayun yang kemudian dikirimkan ke Arduino untuk dianalisa. Bentuk perangkat, pemasangan *encoder* posisi ayun pada troli, mekanisme kerja troli terhadap motor DC, mekanisme kerja troli terhadap pulley, dan pemasangan batang ayun pada poros *encoder* posisi ayun dapat dilihat pada Gambar 3-25 hingga 3-28.



Gambar 3 - 25 Pemasangan Enkoder Posisi Ayun Pada Troli.



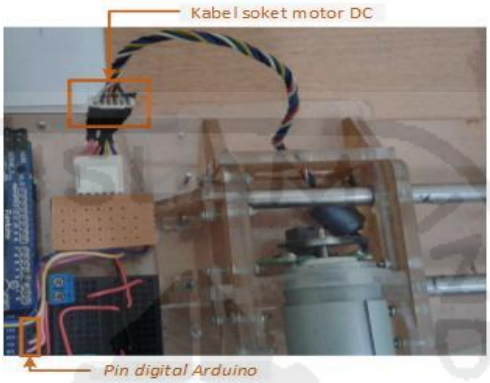
Gambar 3 - 26 Mekanisme Kerja Troli Terhadap Motor DC.



Gambar 3 - 27 Mekanisme Kerja Troli Terhadap Pulley.

Gambar 3 - 28 Pemasangan Batang Ayun Pada Enkoder Posisi Ayun.

Tahap berikutnya adalah menghubungkan kabel *channel* A dan B *encoder* motor DC ke pin *digital* yang terdap pada papan Arduino. Kemudian hubungkan kebel Vcc dan Gnd ke pin papan Arduino yang sudah tertera tulisan Vcc dan Gnd. Pemasangan kabel antara motor DC pada Arduino dapat dilihat pada Gambar 3-29 dan Tabel 3-3 menunjukan hubungan pin *encoder* motor DC dan Arduino.



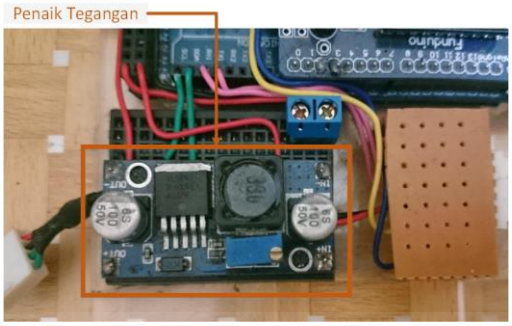
Gambar 3 - 29 Pengkabelan Enkoder dan Vcc serta Ground Motor DC Pada Pin Digital Arduino.

Tabel 3 - 3 Hubungan Pin Enkoder Motor DC dengan Pin Digital Arduino.

Warna Kabel Motor DC	Fungsi Warna	Pin Arduino
Merah	Vcc	Vcc
Hitam	Gnd	Gnd
Putih	*Channel* A	19
Hijau	*Channel* B	18

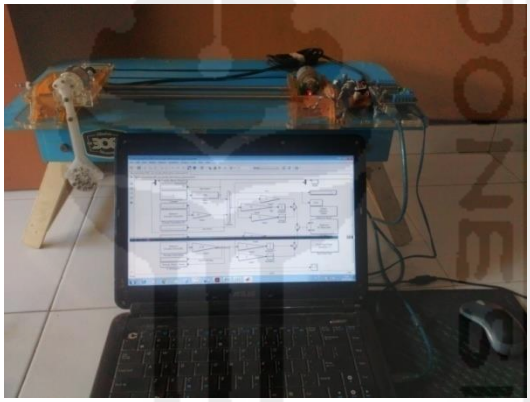
Pemasangan *encoder* posisi ayun ke Arduino membutuhkan suplai tegangan 24 Volt sedangkan Arduino hanya dapat memberikan suplai tegangan hingga 5 volt. Solusi dari masalah tersebut yaitu dengan memasang penaik tegangan. Penaik tegangan yang digunakan dalam penelitian ini dapat menaikan tegangan dari 3,3 volt hingga 35 volt. Pemasangan penaik tegangan *encoder* posisi ayun ke Arduino ditunjukan pada Gambar 3-30. Tegangan 5 volt dari Arduino dihubungkan ke *input* positif penaik tegangan dan *ground* Arduino dihubungkan ke *input* negatif penaik tegangan. Untuk *output* dari penaik tegangan dihubungkan ke pin daya *encoder* posisi ayun, kemudian *channel* A dan

B *encoder* posisi ayunan langsung dihubungkan ke pin 20 dan pin 21 yang terdapat pada papan Arduino.

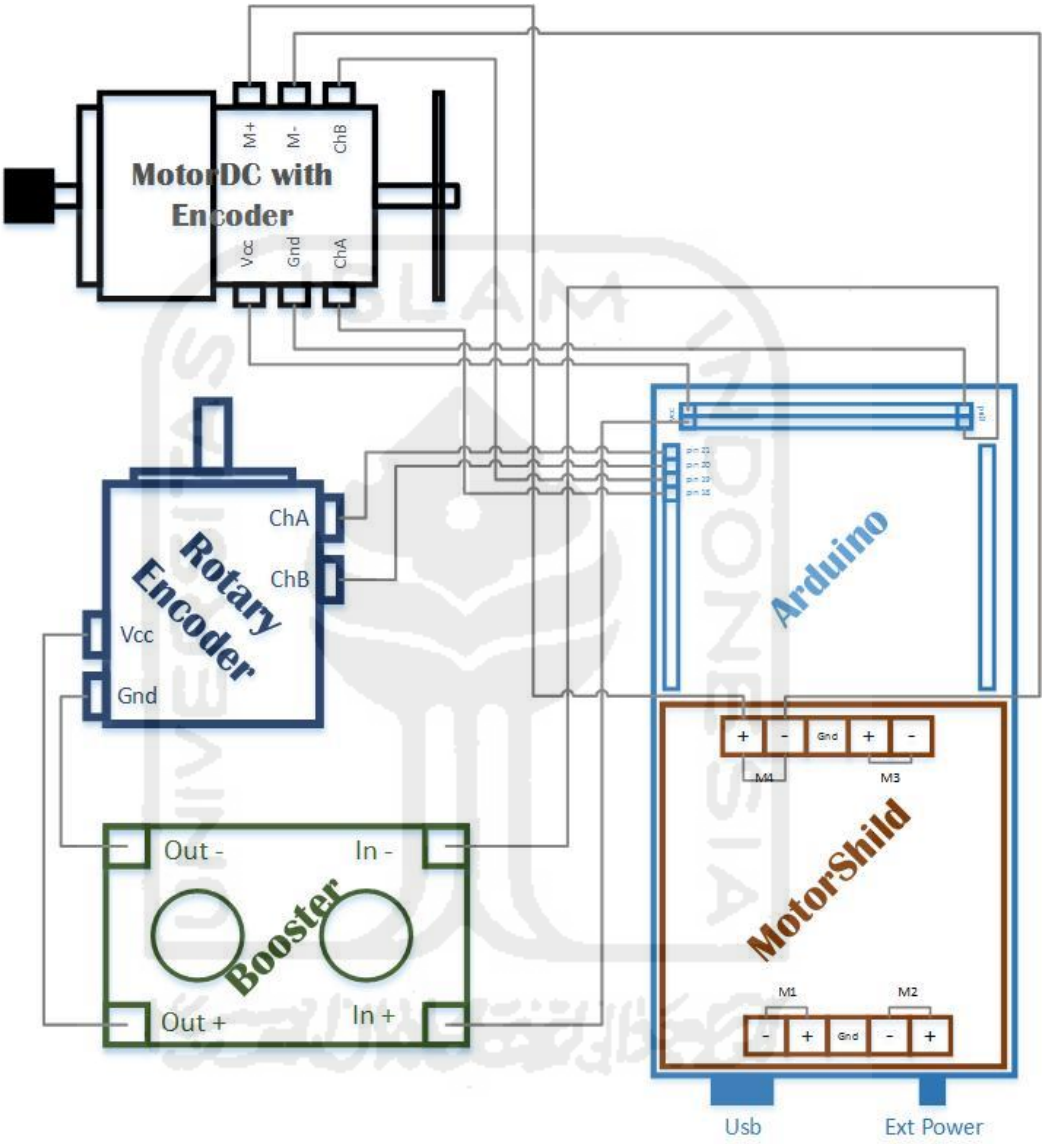


Gambar 3 - 30 Instalasi Penaik Tegangan Enkoder Posisi Ayun.

Hasil perancangan perangkat keras seluruhnya dapat dilihat pada gambar 3-31 dan skema perkabelan perangkat keras pada Gambar 3-32 dibawah ini.



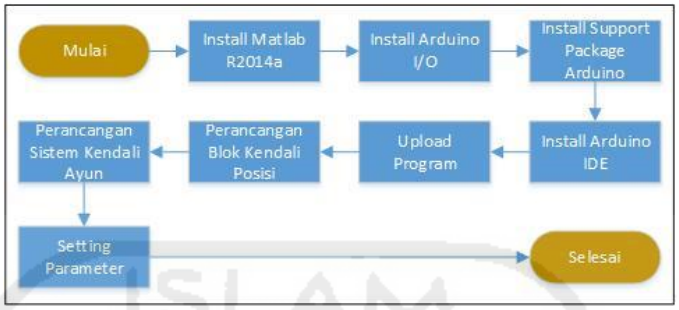
Gambar 3 - 31 Hasil Perancangan Perangkat Keras.



Gambar 3 - 32 Skema perkabelan simulator *crane*.

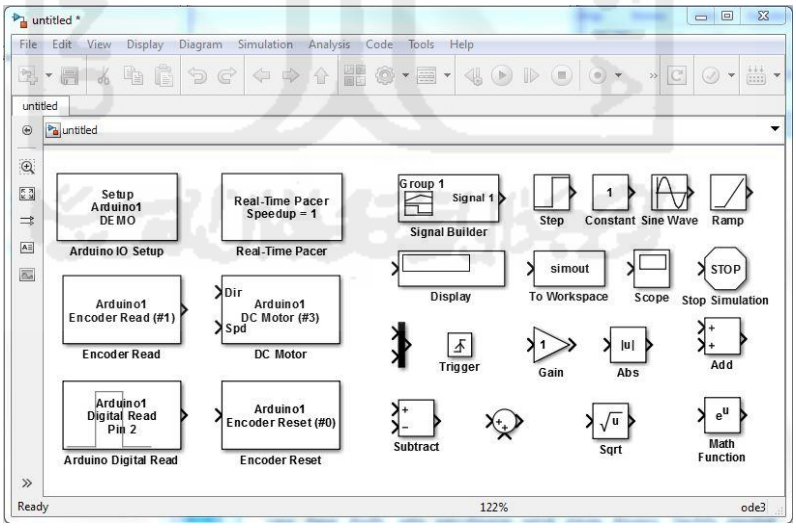
3.3 Perancangan Perangkat Lunak

Dalam instalasi perangkat lunak ada beberapa tahapan yang harus dilakukan seperti yang ditunjukan pada Gambar 3-33.



Gambar 3 - 33 Proses Perancangan Perangkat Lunak.

Perangkat lunak analisis yang digunakan dalam penelitian ini adalah Matlab Simulink. Simulink merupakan *diagram block environtment* yang dapat digunakan untuk simulasi, pembuatan program otomatis, verifikasi model, dan juga pengujian kontinyu. Simulink juga menyediakan fitur editor grafis, librari yang dapat diedit, serta penyelesaian untuk sistem dinamis. Jendela Simulink dapat dilihat pada Gambar 3-34. Penelitian ini dilakukan menggunakan sistem operasi Windows 7 Ultimate 32bit.



Gambar 3 - 34 Tampilan Matlab Simulink R2014a.

3.4 Perancangan Sistem Kendali Simulator *Crane*

Perancangan sistem kendali *crane* terdiri dari dua tahap, yaitu perancangan sistem kendali posisi motor DC yang menggerakan simulator troli dan kemudian digabungkan dengan sistem kendali posisi ayun simulator *crane*.

3.4.1 Perancangan Sistem Kendali Posisi Simulator *Crane*

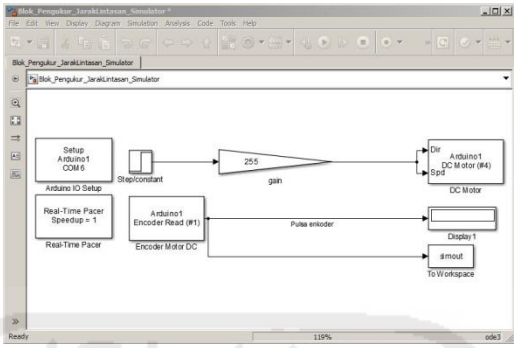
Kontrol posisi motor DC adalah suatu proses mengendalikan sebuah objek yaitu posisi simulator troli untuk mencapai posisi yang diharapkan, yang mana nilai *input*-nya disesuaikan oleh operator pengendali. Pengujian ini dilakukan dengan menggunakan kontroler PID (Proporsional, Integral, Derivatif), untuk menggunakan kontroler PID ini maka harus diketahui terlebih dahulu referensi panjang lintasan yang akan ditempuh oleh troli selama pengujian berlangsung. Motor DC terhubung dengan enkoder berfungsi sebagai pembacaan nilai aktual pada saat troli bergerak.

Untuk mengatasi permasalahan tersebut maka dilakukan pengukuran panjang lintasan yang akan dilalui oleh troli akibat putaran poros motor DC. Pengukuran panjang lintasan troli dilakukan secara manual dengan menggunakan bantuan alat ukur seperti penggaris, titik panjang lintasan yang akan diukur berada pada sisi troli terluar hingga kebidang ujung lintasan. Pengukuran panjang lintasan troli dapat dilihat pada Gambar 3-35 dibawah ini.

Gambar 3 - 35 Pengukuran panjang lintasan troli dengan alat ukur penggaris.

Dari pengukuran jarak lintasan troli yang telah ditunjukan pada Gambar 3-35 didapatlah hasil pengukuran panjang lintasan yaitu sebesar ±250 milimeter

yang kemudian hasil dari pengukuran panjang lintasan troli ini diolah kedalam susunan blok Simulink yang telah disusun.



Gambar 3 - 36 Blok simulink untuk mencari total panjang lintasan simulator yang kemudian nilai pulsa dikonversi menjadi milimeter.

Pada Gambar 3-36 digunakan blok Simulink yang tersusun komponen untuk mencari jumlah pulsa panjang lintasan, komponen blok Simulink yang digunakan adalah blok *step* sebagai masukan, blok *gain* bernilai 255 agar putaran poros motor DC bernilai besar dan mampu menggerakan troli dengan cepat hingga mencapai ujung lintasan juga menghasilkan nilai pulsa yang diinginkan, blok DC motor sebagai blok yang mengkomunikasikan Matlab Simulink dengan perangkat kerasnya, disisi lain ada blok pembacaan enkoder (*encoder read*) yang memberikan nilai aktual berupa pulsa yang ditunjukan oleh blok *display*.

Tabel 3 - 4 Pengujian mencari jumlah pulsa jarak lintasan troli.

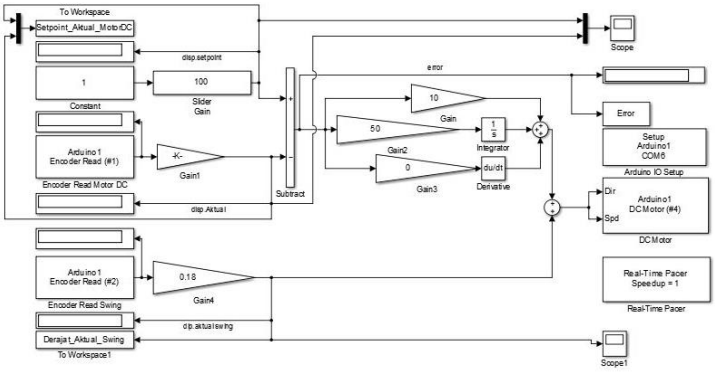
Pengujian	**Panjang Lintasan (mm)**	**Pembacaan Encoder Motor DC (Pulsa)**	**Nilai Pengali Konversi Pulsa ke mm**	**Nilai Pengali Konversi mm ke Pulsa**
1	±250	16262	0,0159	63,77
2	±250	16246	0,0157	63,71
3	±250	16265	0,0157	63,78
4	±250	16264	0,01568	63,78
5	±250	16257	0,0157	63,75
Rata-Rata		16259	0,0157	63,76

Pada Tabel 3-4 diketahui total panjang lintasan troli sebesar ±250 milimeter dan rata-rata jumlah pulsa yang dihasilkan enkoder motor DC dari hasil 5 kali pengujian didapat nilai sebesar 16259 pulsa.

Maka untuk dapat menghasilkan keluaran nilai aktual dalam satuan milimeter dari pembacaan enkoder yang keluarannya pulsa dapat dilihat pada perhitungan dibawah ini :

$$\frac{\text{Panjang Lintasan (mm)}}{\text{Jumlah Pulsa Panjang Lintasan (Pulsa)}} = \frac{250\ \text{mm}}{16259\ P} = 0{,}0157\ \text{mm/P}$$

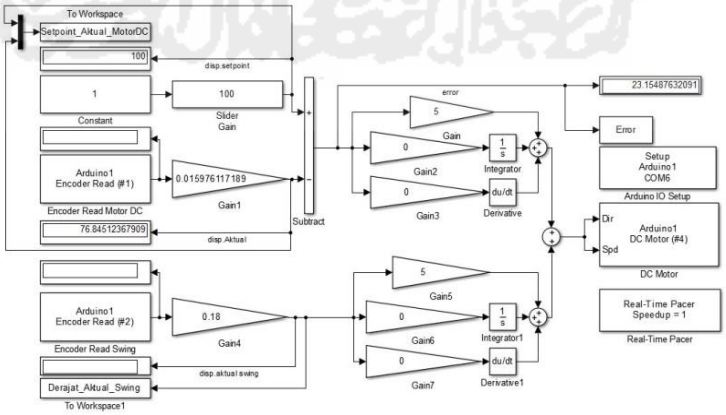
Dari perhitungan yang telah dilakukan diatas, didapat hasil 0,0157 mm/pulsa. Nilai tersebut kemudian dimasukan kedalam blok *gain*1 agar keluaran dari *encoder read* yang bernilai pulsa dapat dikonversikan menjadi milimeter dengan mengkalikan nilai pulsa enkoder dengan nilai blok *gain*1. Jadi, setiap 1 pulsa yang keluar menghasilkan nilai konversi 0,0157 mm begitupun seterusnya.
Blok Simulink sistem kendali posisi motor DC yang menggerakan simulator troli terdiri dari blok step/*constant* dan blok *slider gain* sebagai *input*, blok *substract* sebagai blok pembanding, blok pembacaan enkoder motor DC, blok kontroler PID, blok *gain*, blok *sum*, dan *blok* motor DC. Blok *input* memiliki *output setpoint*/posisi target motor DC. Blok pembacaan enkoder motor DC dihubungkan ke blok *gain* guna mengkonversi nilai pulsa yang dihasilkan blok pembacaan enkoder menjadi nilai satuan milimeter. Kemudian Blok *input* dan blok pembacaan enkoder motor DC dihubungkan ke blok *substract* guna membaca *error* yang terjadi. Blok *substract* dihubungkan ke blok kontroler PID guna memberikan kompensasi untuk *error* yang muncul, blok kontroler PID yang digunakan tidak dalam satu blok utuh akan tetapi dibuat blok terpisah seperti untuk kontrol P (proporsional) hanya menggunakan blok *gain*, kemudian blok *gain* ditambah blok *integrator* sebagai kontrol I (integral), lalu blok *gain* ditambah blok *derivative* sebagai kontrol D (derivatif), kemudian ketiga blok tersebut dihungkan ke blok *sum* bertanda + (plus) guna mengakumulasikan keluaran dari ketiga blok tersebut. Blok *sum* selanjutnya dihubungkan ke blok DC motor. Pembacaan ayunan batang ayun saat simulator troli bergerak ditampilkan oleh blok pembacaan enkoder posisi ayunan melalui *scope*. Untuk lebih jelasnya dapat dilihat pada Gambar 3-37.



Gambar 3 - 37 Susunan Blok Sistem Kendali Posisi Simulator Troli.

3.4.2 Perancangan Blok Sistem Kendali Posisi Ayun

Sistem kendali posisi ayun dengan sistem kendali posisi troli merupakan susunan blok sistem kendali posisi yang diberikan penambahan beberapa blok. Blok baru yang ditambahkan yaitu blok kontroler PID, blok *gain*, blok pembacaan enkoder ayunan, dan blok *sum*. Blok pembacaan ayunan dihubungkan ke blok *gain* guna mengkonversi nilai pulsa yang dihasilkan blok pembacaan enkoder ayun menjadi nilai satuan derajat. Kemudian blok *gain* dihubungkan ke blok PID, blok PID kendali ayun kondisinya sama seperti susunan blok PID yang ada di kendali posisi motor DC. Kemudian blok PID dihubungkan ke blok *sum*. Blok *sum* ayun dan blok *sum* kendali motor DC dihubungkan kembali ke blok *sum* guna memberikan kompensasi *error* yang muncul. Blok *sum* terakhir dihubungkan ke blok DC motor untuk diolah kembali. Susunan blok kendali ayun dapat dilihat pada Gambar 3-38.



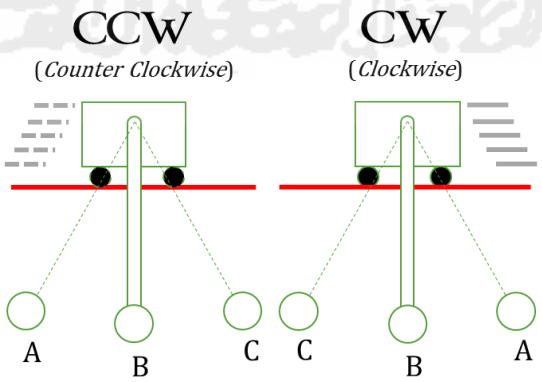
Gambar 3 - 38 Blok Sistem Kendali Ayun dengan Kontroler.

Untuk kendali posisi ayun, pada pembacaan enkoder ayun yang keluarannya pulsa juga harus dikonversikan menjadi satuan derajat karena ayunan bergerak secara rotasi.

Untuk mengkonversi nilai pulsa menjadi satuan derajat, maka dilakukan pengujian untuk mengukur putaran poros *incremental encoder* ayun untuk mengetahui jumlah pulsa dalam satu kali putaran. Dalam satu kali putaran, poros berputar 360°, dalam satu kali putaran poros menghasilkan pulsa sebanyak 500. Karena *incremental encoder* yang digunakan bertipe enkoder *quadrature*, maka setiap pulsa yang dihasilkan akan dikalikan 4. *Sehingga* dalam satu kali putaran enkoder ayun menghasilkan 500 x 4 = 2000 pulsa. Maka untuk dapat menghasilkan keluaran nilai aktual dalam satuan derajat dari pembacaan enkoder yang keluarannya pulsa dapat dilihat pada perhitungan dibawah ini :

$$\frac{\text{1 Putaran Poros (Derajat)}}{\text{Jumlah Pulsa 1 Putaran Poros (Pulsa)}} = \frac{360°}{2000\ \text{P}} = 0{,}18°/\text{P}$$

Dari perhitungan yang telah dilakukan diatas, didapat hasil 0,18°/pulsa. Nilai tersebut kemudian dimasukan kedalam blok *gain* agar keluaran dari *encoder read* kendali ayun yang bernilai pulsa dapat dikonversikan menjadi derajat dengan mengkalikan nilai pulsa enkoder dengan nilai blok *gain*. Jadi, setiap 1 pulsa yang keluar menghasilkan nilai konversi 0,18° begitupun seterusnya. Posisi batang ayun sebagai gambaran umum dapat dilihat pada Gambar 3-39 dibawah ini.



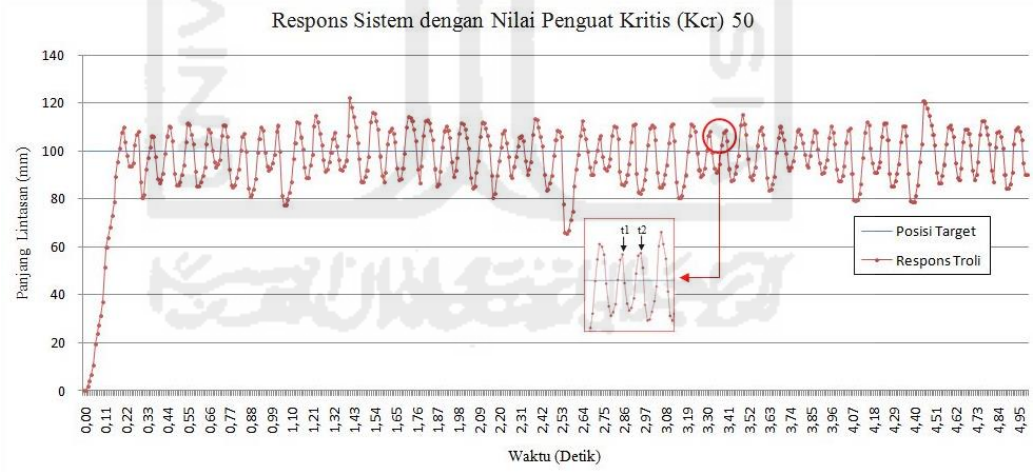
Gambar 3 - 39 Gambaran posisi batang ayun (posisi A batang ayun berayun kebelakang, Posisi B batang ayun tetap, Posisi C batang ayun berayun kedepan)

Berdasarkan gambar 3-39 diatas tersebut posisi awal batang ayun berada pada huruf/posisi B. Jika troli bergerak dari arah cw (*clockwise*), kondisi batang ayun pada posisi A berayun kebelakang, jika pada batang ayun pada posisi C berayun kebelakang.

3.4.3 Perancangan Kontroler PID dengan Metode Ziegler-Nichols 2

Perancangan kontroler PID dengan metode Ziegler-Nichols 2 merupakan perancangan kontroler dengan melihat respons sistem. Perancangan dengan menggunakan metode Ziegler-Nichols 2 ini mula-mula periode integral (T_i) dan periode derivatif (T_d) diberikan nilai 0. Nilai penguat dinaikan sedikit demi sedikit hingga respons sistem berosilasi secara kontinyu atau terus menerus. Periode osilasi sistem kemudian digunakan untuk mencari nilai K_i dan K_d.

Pada pengujian ini nilai penguat kritis (K_{cr}) 50 didapat respons sistem berosilasi secara kontinyu. Dari grafik tersebut maka dapat ditentukan nilai P_{cr}-nya untuk menentukan nilai T_i dan T_d. Lihat Tabel 2-2 untuk mencari nilai P_{cr}, K_p, K_i dan K_d. Grafik osilasi sistem dapat dilihat pada Gambar 3-40.



Gambar 3 - 40 Grafik Osilasi Sistem dengan Nilai Penguat Kritis 50.

Mencari Nilai P_{cr} :

$P_{cr} = (t_2 - t_1) \times$ *Sample time*

$P_{cr} = (108{,}5634 - 108{,}0772) \times$ *Sample time*

$P_{cr} = 0{,}4862 \times 1$ *Second* $= 0{,}4862$

Mencari Nilai K_p :

$K_p = 0{,}6 \times (K_{cr}) = 0{,}6 \times 50 = 30$

Setelah menghitung nilai P_{cr} kemudian menghitung nilai T_i dan T_d :

$T_i = 0{,}5 \times (P_{cr}) = 0{,}5 \times 0{,}4862 = 0{,}2431$

$T_d = 0{,}125 \times (P_{cr}) = 0{,}125 \times 0{,}4862 = 0{,}060775$

Nilai T_i dan T_d selanjutnya digunakan untuk mencari nilai K_i dan K_d melalui Persamaan 2-3 dan 2-4. Maka nilai K_i dan K_d menjadi :

$K_i = \frac{30}{0{,}2431} = 123{,}4$

$K_d = 30 \times 0{,}060775 = 1{,}8$

Setelah mencari melalui persamaan maka didapatkan nilai kontroler PID kendali posisi troli dengan menggunakan metode Ziegler-Nichols 2 adalah K_p=30, K_i=123,4, dan K_d=1,8. Untuk nilai kontroler kendali posisi ayun disamakan dengan kontroler kendali posisi troli.

3.4.4 Perancangan Kontroler PID dengan Metode *Trial and Error*

Perancangan kontroler PID menggunakan metode *trial and error* merupakan salah satu cara mencari nilai K_p, K_i, dan K_d dengan melakukan pengujian berulang kali untuk mencapai nilai yang diinginkan. Pada perancangan menggunakan metode ini hanya perlu mengatur setiap variabel kontroler PID hingga respons sistem mencapai karakteristik yang diinginkaan.

Pada pengujian ini digunakan nilai K_p 1 dengan K_i 40-45, K_p 2 dengan K_i 40-45, K_p 3 dengan K_i 10-15, K_p 4 dengan K_i 30-40, K_p 5 dengan K_i 10-15 dan 40-50, K_p 6 dengan K_i 1-10 dan 30-35, K_p 7 dengan K_i 15-25 dan 30-40, K_p 8 dengan K_i 10-20 dan 45-50, dan K_p 9 dengan K_i 5-15. Rentang nilai tersebut dipilih karena pada rentang tersebut kompensasi yang diberikan mulai berpengaruh pada sistem. Nilai K_d pada pengujian ini tidak dicari, karena nilai K_d tidak berpengaruh besar pada respons sistem pada pengujian ini.

Respons sistem yang dicari sesuai karakteristik yang diinginkan dengan metode ini yaitu tidak memiliki maksimal *overshoot* (M_p), M_p yang diijinkan tidak boleh lebih dari 10%, jika banyak respons masuk kedalam nilai 10% yang diijinkan, maka dipilih nilai M_p terkecil atau yang mendekati nilai posisi target;

error steady state (e_{ss}) yang diijinkan tidak melebihi 2% batas atas yaitu 102 mm dan tidak kurang -2% batas bawah yaitu 98 mm dari posisi target yang telah ditentukan yaitu 100 mm; dan memiliki pencapaian *settling time* (t_s) kurang dari 3,75 detik. Respons sistem dengan kontroler PID metode *trial and error* dapat dilihat pada Tabel 3-6 yang menunjukan nilai M_p, t_s, dan e_{ss} menggunakan kontroler PID metode *trial and error*. Beberapa respons sistem memenuhi kriteria M_p dan t_s seperti yang ditentukan, beberapa respons tersebut kemudian dipilih yang memiliki *error steady state* (e_{ss}) terkecil. Agar lebih jelasnya Tabel 3-5 dibawah ini menunjukan kriteria respons yang diinginkan dengan kriteria respons yang baik.

Tabel 3 - 5 Kriteria respons sistem yang diinginkan dengan kriteria respons sistem yang baik pada karakteristik respons sistem transien

Karakteristik Respons Sistem transien	**Kriteria respons yang diinginkan**	**Kriteria respons yang baik**
Settling time, ts	< 3,75 detik	< 5detik
Maksimal *Overshoot*, Mp	< 5% atau < 5mm	< 10% atau < 10mm
Error Steady State, Ess	Posisi steady state mendekati target posisi	-2% dan +2% dari target posisi
Ayunan	Mereduksi hingga < 5˚ (derajat)	Mereduksi hingga mencapai 5˚(derajat)

Tabel 3 - 6 Nilai M_p, E_{ss}, dan M_p Menggunakan Kontroler PID Metode *Trial and Error*

K_i	K_p	1				K_i	K_p	2				K_i	K_p	3				K_i	K_p	4				K_i	K_p	5			
	Posisi Target (mm)	Posisi Steady State (mm)	Settling Time (detik)	Maksimal Overshoot (mm)	Error Steady State (mm)		Posisi Target (mm)	Posisi Steady State (mm)	Settling Time (detik)	Maksimal Overshoot (mm)	Error Steady State (mm)		Posisi Target (mm)	Posisi Steady State (mm)	Settling Time (detik)	Maksimal Overshoot (mm)	Error Steady State (mm)		Posisi Target (mm)	Posisi Steady State (mm)	Settling Time (detik)	Maksimal Overshoot (mm)	Error Steady State (mm)		Posisi Target (mm)	Posisi Steady State (mm)	Settling Time (detik)	Maksimal Overshoot (mm)	Error Steady State (mm)
40	100	112,57	-	19,87	12,57	40	100	108,59	-	8,62	8,59	10	100	94,30	-	-	-5,70	30	100	100,50	2,72	0,50	0,50	10	100	94,46	-	-	-5,54
41	100	113,80	-	16,95	13,80	41	100	113,99	-	14,11	13,99	11	100	97,49	-	-	-2,51	31	100	100,39	3,42	0,39	0,39	11	100	92,70	-	-	-7,30
42	100	99,85	4,93	19,76	-0,15	42	100	108,84	-	8,89	8,84	12	100	98,13	4,97	-	-1,87	32	100	100,34	4,08	0,34	0,34	12	100	92,17	-	-	-7,83
43	100	98,91	4,66	19,88	-1,09	43	100	111,35	-	11,38	11,35	13	100	98,03	4,85	-	-1,97	33	100	100,51	2,49	0,51	0,51	13	100	92,14	-	-	-7,86
44	100	111,59	-	11,96	11,59	44	100	111,38	-	11,51	11,38	14	100	100,50	4,38	0,50	0,50	34	100	100,53	3,07	0,53	0,53	14	100	92,94	-	-	-7,06
45	100	99,90	4,94	17,62	0,10	45	100	108,84	-	8,86	8,84	15	100	100,70	3,88	0,70	0,70	35	100	100,53	2,58	0,53	0,53	15	100	94,74	-	-	-5,26
																		36	100	99,01	0,17	-	-0,99	40	100	98,25	4,95	-	-1,75
																		37	100	100,58	2,28	0,58	0,58	41	100	100,29	3,82	0,29	0,29
																		38	100	100,59	2,12	0,59	0,59	42	100	98,94	4,30	-	-1,06
																		39	100	103,08	-	3,08	3,08	43	100	97,99	-	-	-2,01
																		40	100	102,96	-	2,96	2,96	44	100	101,05	0,19	1,05	1,05
																								45	100	100,09	3,91	0,09	0,09
																								46	100	103,19	-	4,32	3,19
																								47	100	102,91	-	5,45	2,91
																								48	100	100,21	1,68	0,21	0,21
																								49	100	100,12	0,20	0,12	0,12
																								50	100	100,23	1,68	0,23	0,23

K_i	K_p	6				K_i	K_p	7				K_i	K_p	8				K_i	K_p	9			
	Posisi Target (mm)	Posisi Steady State (mm)	Settling Time (detik)	Maksimal Overshoot (mm)	Error Steady State (mm)		Posisi Target (mm)	Posisi Steady State (mm)	Settling Time (detik)	Maksimal Overshoot (mm)	Error Steady State (mm)		Posisi Target (mm)	Posisi Steady State (mm)	Settling Time (detik)	Maksimal Overshoot (mm)	Error Steady State (mm)		Posisi Target (mm)	Posisi Steady State (mm)	Settling Time (detik)	Maksimal Overshoot (mm)	Error Steady State (mm)
1	100	89,00	-	-	-11,00	15	100	96,43	-	-	-3,57	10	100	105,63	-	7,26	5,63	5	100	105,53	-	6,58	5,53
2	100	91,56	-	-	-8,44	16	100	96,50	-	-	-3,50	11	100	100,98	0,17	0,98	0,98	6	100	100,98	0,12	1,01	0,98
3	100	101,78	0,17	1,78	1,78	17	100	97,60	-	-	-2,40	12	100	104,18	-	5,14	,4,18	7	100	105,56	-	5,78	5,56
4	100	92,23	-	-	-7,77	18	100	103,38	-	3,79	3,38	13	100	102,19	-	2,21	2,19	8	100	105,67	-	20,26	5,67
5	100	91,73	-	-	-8,27	19	100	97,23	-	-	-2,77	14	100	105,56	-	6,52	5,56	9	100	105,45	-	5,86	5,45
6	100	91,84	-	-	-8,16	20	100	101,45	0,21	1,45	1,45	15	100	105,37	-	6,61	5,37	10	100	105,56	-	10,49	5,56
7	100	111,37	-	14,47	11,37	21	100	103,16	-	3,90	3,16	16	100	103,18	-	4,53	3,18	11	100	105,67	-	11,41	5,67
8	100	92,06	-	-	-7,94	22	100	100,61	0,18	0,64	0,61	17	100	103,08	-	3,52	3,08	12	100	105,59	-	8,39	5,59
9	100	91,92	-	-	-8,08	23	100	103,04	-	3,21	3,04	18	100	103,10	-	3,30	3,10	13	100	105,72	-	9,03	5,72
10	100	94,99	-	-	-5,01	24	100	103,16	-	3,49	3,16	19	100	105.37	-	9,70	5,37	14	100	101,49	0,15	1,49	1,49
30	100	98,03	0,20	-	-1,97	25	100	101,99	0,17	1,99	1,99	20	100	105,45	-	6,24	5,45	15	100	105,48	-	13,40	5,48
31	100	97,61	-	-	-2,39	30	100	105,36	-	5,78	5,36	45	100	102,60	-	10,61	2,60						
32	100	98,58	0,19	-	-1,42	31	100	103,07	-	3,15	3,07	46	100	102,43	-	8,75	2,43						
33	100	99,82	0,21	-	-0,18	32	100	102,77	-	9,56	2,83	47	100	100,51	4,29	17,59	0,51						
34	100	100,29	0,20	0,29	0,29	33	100	102,77	-	11,85	2,77	48	100	100,28	4,30	16,10	0,28						
35	100	100,00	0,19	-	0,00	34	100	98,38	0,16	-	-1,62	49	100	102,30	-	8,56	2,30						
						35	100	101,89	0,14	1,91	1,89	50	100	101,49	4,61	8,70	1,49						
						36	100	103,24	-	3,60	3,24												
						37	100	102,39	-	12,71	2,39												
						38	100	102,97	-	5,03	2,97												
						39	100	102,94	-	4,43	2,94												
						40	100	102,82	-	5,99	2,82												

Tabel 3-5 diatas menunjukan respons sistem posisi *steady state*, M_p (maksimal *overshoot*), t_s (*settling time*), dan e_{ss} (*error steady state*) berdasarkan hasil pengujian pada variabel kontroler yaitu K_p dan K_i. Hasil pengujian pada setiap nilai variabel kontroler tersebut menghasilkan data berbentuk kurva, kemudian data berbentuk kurva tersebut dianalisis sesuai karakteristik sistem yang telah ditentukan.

Respons sistem dipilih yang memiliki pencapaian t_s (*settling time*) < 3,75 detik yaitu berada pada K_p 4 dengan K_i 30,31,33-38, K_p 5 dengan K_i 44, 48-50, K_p 6 dengan K_i 3, 30, 32-35, K_p 7 dengan K_i 20, 22, 25, 34, 35, K_p 8 dengan K_i 11, dan K_p 9 dengan K_i 6, 14. Variabel kontroler yang masuk kedalam karakteristik t_s yang diijinkan kemudian dipilih respons sistem yang memiliki M_p (maksimal *overshoot*) < 10 % atau 10 mm, jika banyak dari respons sistem M_p < 10 %, maka dipilih nilai terkecil saja. Karena respons sistem M_p (maksimal overshoot) pada variabel kontroler yang dipilih sesuai t_s yang diinginkan memiliki rata-rata maksimal *overshoot* dibawah 2 % atau 2 mm, maka dipilih nilai variabel kontroler yang memiliki maksimal *overshoot* < 0,5 mm dan yang tidak memiliki maksimal *overshoot* yaitu berada pada K_p 4 dengan K_i 31 dan 36, K_p 5 dengan K_i 48-50, K_p 6 dengan K_i 30, 32-35, dan K_p 7 dengan K_i 34. Setelah mendapatkan variabel kontroler yang memiliki nilai t_s dan M_p sesuai dengan karakteristik yang diinginkan, kemudian pilih respons sistem yang memiliki nilai *error steady state* (e_{ss}) yang masuk kedalam batas atas dan bawah yang diijinkan yaitu 2%, kemudian pilih nilai yang memiliki e_{ss} terkecil atau mendekati nilai posisi target. Nilai e_{ss} (*error steady state*) terkecil berada pada K_p 6 dengan K_i 33 yang mempunyai nilai ess sebesar -0,18 mm, dan K_p 6 dengan K_i 35 yang mempunyai nilai e_{ss} sebesar 0,00 mm atau tepat berada pada posisi target yang ditentukan.

Berdasarkan analisis respons sistem diatas maka dipilihlah nilai variabel kontroler PID metode *trial and error* yaitu K_p 6 dengan K_i 33. Untuk nilai K_p 6 dengan K_i 35 tidak dipilih karena respons tersebut memiliki nilai e_{ss} 0,00 mm, setidaknya dalam respons sistem yang dihasilkan harus ada angka yang muncul walaupun bernilai 0,01 mm. Untuk nilai variabel kontroler kendali posisi ayun disamakan dengan variabel kontroler kendali posisi troli.

Untuk mendukung penyajian data hasil respons sistem nilai posisi *steady* state, M_p (maksimal *overshoot*), e_{ss} (*error steady* state), dan ts (*settling time*) yang telah ditunjukan pada Tabel 3-5 diatas, disajikan juga respons sistem posisi *steady state*, Mp, ess, dan ts dalam bentuk grafik dari masing-masing variabel kontroler, guna memudahkan dalam pengamatan respons sistem. Grafik respons sistem tersebut dapat dilihat pada halaman lampiran 1 yang terdapat di halaman terakhir tugas akhir ini.