

Pembuatan Lengan Robot Dengan Sistem Pemrograman Menggunakan *Gesture Control*

Hans Prasetya Cokro, Felix Pasila, Petrus Santoso
Program Studi Teknik Elektro, Universitas Kristen Petra
Jl.Siwalankerto 121-131, Surabaya 60236, Indonesia
E-Mail: m23412041@john.petra.ac.id

Abstrak- Saat ini dibutuhkan peran otomasi dalam setiap industri. Beberapa permasalahan disebabkan karena peningkatan Upah Minimum Regional, dan susah nya melakukan kontrol terhadap para pekerja.

Lengan robot ini akan mempermudah proses pemrograman robot industri agar robot industri dapat diterima di berbagai perusahaan di Indonesia. Lengan robot ini menggunakan mekanik *sling* dan memiliki kemampuan hanya untuk proses *pick and place*, dengan kapasitas 4 axis, dan metode pengajaran melalui *kinesthetic teaching* dengan *gesture control*. Lengan robot memanfaatkan *myo armband* serta sensor *electromyograph* untuk membaca gerakan lengan operator.

Kata Kunci- *Gesture control*, lengan robot, *kinetic teaching*, *rotary encoder*

I. PENDAHULUAN

Indonesia adalah negara dinamis yang memiliki potensi investasi infrastruktur, industri manufaktur, dan jasa yang besar, sehingga untuk bersaing di kawasan regional maupun internasional memerlukan optimalisasi teknologi [1]. Negara Indonesia sedang membutuhkan peran otomasi industri dalam proses produksinya, beberapa disebabkan karena peningkatan Upah Minimum Regional, dan susah nya melakukan proses kontrol terhadap para pekerja [2].

Beberapa industri secara global telah menggunakan lengan robot dalam proses produksinya, terutama dalam industri otomotif yang membutuhkan tingkat presisi yang tinggi [3]. Sistem pemrograman lengan robot yang telah hadir di pasaran menggunakan metode program dengan memanfaatkan *Human machine interface* [4], dan yang terbaru menggunakan metode program dengan memanfaatkan sistem kinetis, yaitu dengan menggerakkan lengan robot pada suatu titik koordinat tujuan, dan menyimpan koordinat yang telah diperoleh, untuk kemudian dapat dilaksanakan robot secara otomatis [5]. Namun untuk metode yang terbaru masih belum dapat digunakan pada robot lengan dengan ukuran yang besar, karena manusia tidak memiliki kemampuan untuk menggerakkan robot lengan dengan ukuran yang besar.

Gesture control merupakan sebuah metode diagnosa, evaluasi, dan rekam aktifitas yang dihasilkan oleh gerakan manusia [6]. Hasil diagnosa dan rekam aktifitas tersebut nantinya yang akan digunakan untuk mengendalikan lengan robot [7]. Metode ini akan menghasilkan pergerakan dari lengan robot sesuai dengan pergerakan lengan operator.

Melihat dari permasalahan itu, maka diperlukan pembuatan lengan robot dengan sistem pemrograman menggunakan *gesture control* sehingga operator akan dimudahkan dalam memprogram gerakan lengan robot baik dalam ukuran kecil maupun besar.

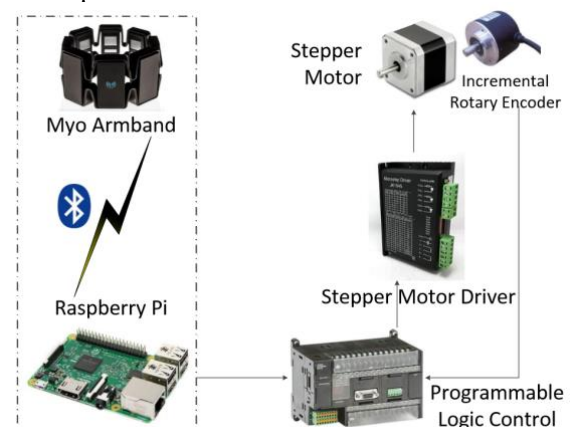
II. PERANCANGAN SISTEM

A. Gambaran Umum Sistem

Sistem pemrograman lengan robot ini adalah sistem *kinesthetic teaching* dan peniruan gerakan lengan manusia. *Input* dari sistem ini adalah *gesture control armband* dan *incremental rotary encoder*, seperti pada gambar 1.

Gesture control armband berfungsi sebagai sensor pembaca gerakan lengan manusia, yang kemudian data yang diperoleh akan diolah oleh *raspberry pi* dan dikirimkan ke *programmable logic control* melalui *digital output* dari *raspberry pi*.

Incremental rotary encoder berfungsi untuk membaca derajat pergerakan masing-masing *joint* daripada lengan robot dan dihubungkan dengan input *programmable logic control* melalui metode *high speed counter*. Setelah itu *programmable logic control* akan menjalankan motor *stepper* yang terdapat pada robot lengan sesuai dengan data yang telah diperoleh



Gambar 1 Desain Sistem Lengan Robot

Sistem *wiring* pada sistem dan gambaran umum pemasangan dapat dilihat pada gambar 2, gambar 3, gambar 4, dan gambar 5. Diagram sistem dapat dilihat pada gambar 1, namun pada gambar 1 hanya ditampilkan sistem untuk satu buah *stepper motor*, dan *incremental rotary encoder*.

B. Desain Panel Robot

Penggerak daripada lengan robot ini terdiri dari enam buah motor *stepper*, lihat gambar 2, namun didalam tugas

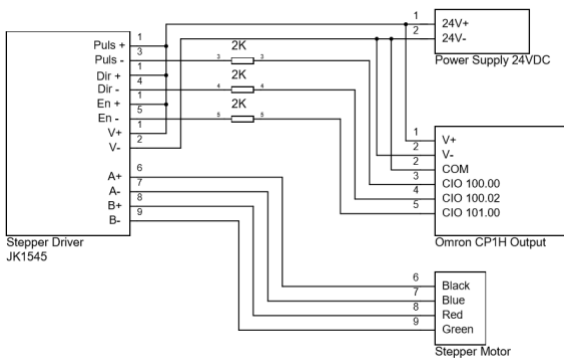
akhir ini hanya akan digunakan empat buah motor *stepper*, yang kemudian motor *stepper* ini dirakit menjadi satu dengan lengan robot, *programmable logic control*, *power supply* dan *incremental rotary encoder*, seperti pada gambar 3 dalam sebuah panel.



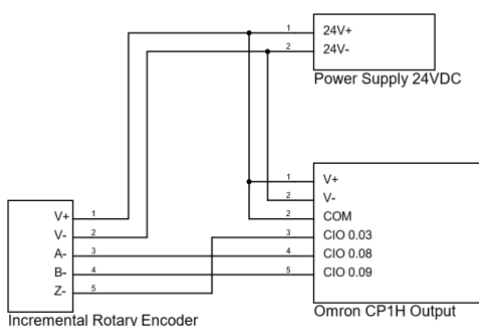
Gambar 2. Gambar susunan panel *stepper motor* dan *controller* daripada lengan robot



Gambar 3 Susunan *incremental rotary encoder* lengan robot



Gambar 4. Wiring diagram antara *stepper motor* dan *controller*

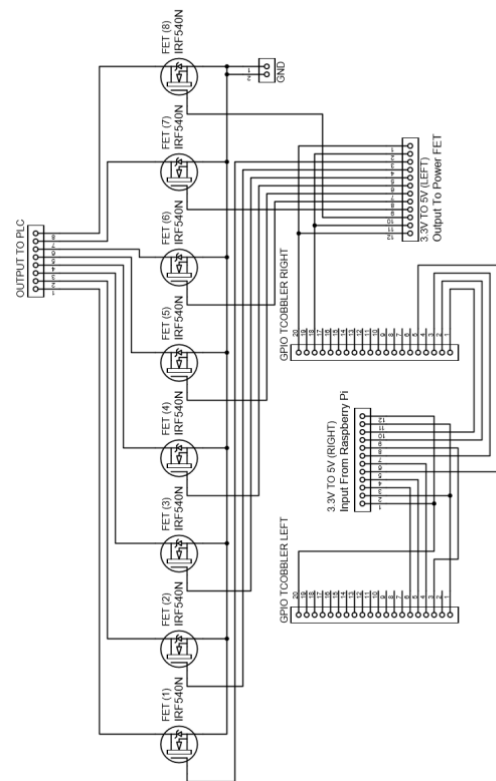


Gambar 5. Wiring diagram *rotary encoder* dengan *controller*

Rangkaian *logic converter* seperti pada gambar 6 dan gambar 7 merupakan rangkaian penghubung antara raspberry pi dengan *programmable logic control*, karena output voltage dari GPIO raspberry pi merupakan 3.3VDC, sedangkan input dari *programmable logic control* hanya dapat aktif apabila menerima supply sebesar 24VDC.

Rangkaian *logic converter* sendiri merubah tegangan dari 3.3VDC menjadi 24VDC. Berikut beberapa komponen yang digunakan dalam desain rangkaian *logic converter* 3.3VDC → 24VDC :

- 1 = T-cobbler plus v1.1 sebagai external gpio raspberry pi
- 2 = 8 channel bi-directional 3.3VDC → 5VDC
- 3 = Power mosfet 5VDC → 24VDC



Gambar 6 Board *logic converter* schematic



Gambar 7 Board *logic converter* yang terdiri dari *power mosfet* dan *bidirectional converter*

C. *Desain Software Pada Raspberry Pi*

Langkah pertama adalah proses *install dependencies* antara *gesture control* dan *raspberry pi*, seperti pada gambar 8. *Dependencies* dilakukan agar *raspberry pi* dapat terkoneksi dengan *gesture control*. Hubungkan dongle

bluetooth dengan usb port pada raspberry pi, dan lakukan hal berikut:

```
// plug bluetooth adapter
// permission to ttyACM0 - must restart linux user after this
sudo usermod -a -G dialout $USER

// dependencies
sudo apt-get install python-pip
sudo pip install pySerial --upgrade
sudo pip install enum34
sudo pip install PyUserInput
sudo apt-get install python-Xlib
sudo apt-get install python-tk

// now reboot
```

Gambar 8 Program *python* untuk melakukan proses *dependencies*

Setelah *dependencies* proses berhasil, selanjutnya dilakukan proses mengunduh library *pyoconnect*, dan meletakkan folder library tersebut ke dalam tempat penyimpanan file yang diinginkan. Langkah ketiga setelah *pyoconnect* berhasil diunduh, digunakan *python* untuk menjalankan *PyoManager.pyc*.



Gambar 9 Tampilan menu utama *pyoconnect*

Langkah keempat yaitu dilakukan uji coba koneksi antara *raspberry pi* dengan *gesture control* dengan menekan tombol *connect myo*, hingga muncul *feedback* melalui terminal *python* yang bertulis *myo connected*.

Langkah kelima adalah pembuatan program untuk mengelolah data *gesture control* untuk mengaktifkan GPIO output,

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(5, GPIO.OUT)
GPIO.setup(6, GPIO.OUT)
GPIO.setup(12, GPIO.OUT)
GPIO.setup(13, GPIO.OUT)
GPIO.setup(16, GPIO.OUT)
```

Gambar 10 Tampilan program *python* GPIO *set up*

Pada gambar 10, merupakan program awal pada *python*, yang berfungsi untuk mendeklarasikan output daripada GPIO *raspberry pi*. Proses deklarasi output dilakukan agar output daripada *raspberry pi* dapat berfungsi seperti yang diinginkan.



Gambar 11 Posisi GPIO pada *raspberry pi*

```
scriptTitle = "RoSys Controller"

def onUnlock():
    myo.unlock("hold")
    print("\n-----Hello Human-----")
    print("-----Welcome to RoSys 1.0-----")
def onPoseEdge(pose, edge):
    if (pose == "waveOut"):
        if (edge == "on"):
            print("\nRobot is going Right/Up")
            GPIO.output(5, GPIO.HIGH)
        if (edge == "off"):
            GPIO.output(5, GPIO.LOW)
    if (pose == "waveIn"):
        if (edge == "on"):
            print("\nRobot is going Left/Down")
            GPIO.output(6, GPIO.HIGH)
        if (edge == "off"):
            GPIO.output(6, GPIO.LOW)
    if (pose == "fist"):
        if (edge == "on"):
            print("\nRobot Vacuum Gripper On")
            GPIO.output(12, GPIO.HIGH)
        if (edge == "off"):
            GPIO.output(12, GPIO.LOW)
    if (pose == "fingersSpread"):
        if (edge == "on"):
            print("\nRobot Vacuum Gripper Off")
            GPIO.output(13, GPIO.HIGH)
        if (edge == "off"):
            GPIO.output(13, GPIO.LOW)
```

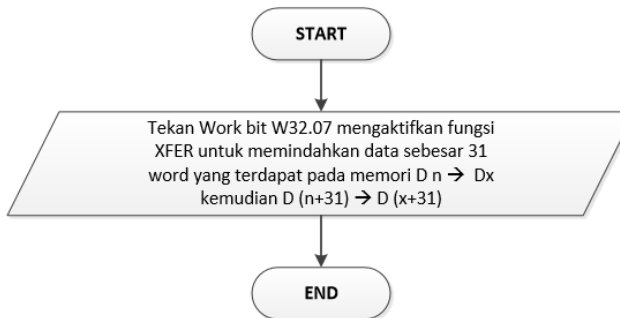
Gambar 12 Program *python* pembacaan *myo armband* dan output yang dihasilkan

```
GPIO.output(13, GPIO.LOW)
if (pose == "rest"):
    if (edge == "on"):
        print("\nHuman Arm Resting, waiting for next command...")
        GPIO.output(5, GPIO.LOW)
        GPIO.output(6, GPIO.LOW)
        GPIO.output(12, GPIO.LOW)
        GPIO.output(13, GPIO.LOW)
```

Gambar 13 Program *python* ketika otot operator dalam keadaan istirahat

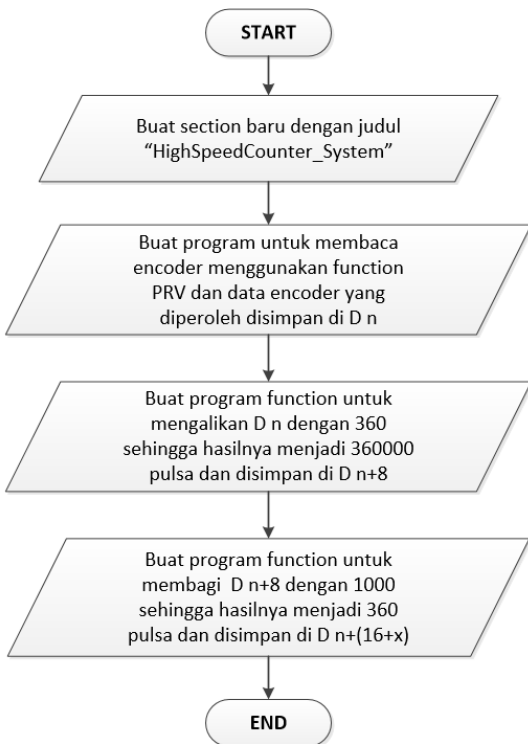
D. *Desain Software Pada Programmable logic control*

Desain *flowchart* menjelaskan secara sekilas proses dari beberapa sistem, dan terbagi menjadi 3 bagian, yaitu *flowchart* untuk membaca *incremental rotary encoder*, inialisasi *home position* untuk *encoder*, proses penerapan keamanan untuk *stepper motor*, proses simulasi robot, dan proses pembacaan *Myo Armband*. Proses lengkap daripada sistem lengan robot akan ditampilkan pada lampiran program.



Gambar 14 *Flowchart* proses menyimpan derajat yang telah di program menuju memori PLC

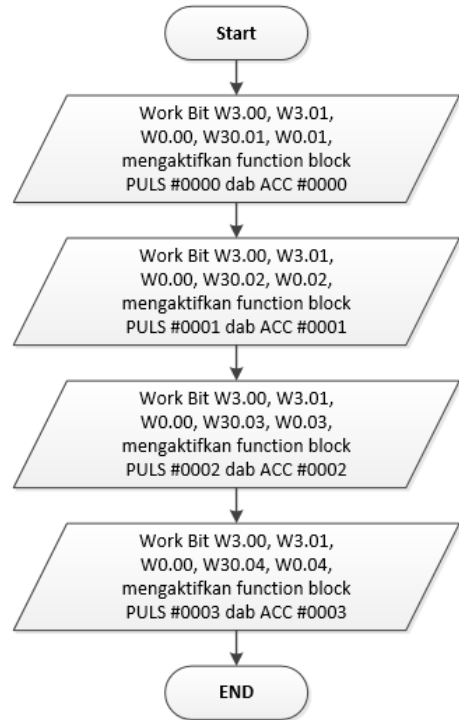
Pada gambar 14, dapat dilihat penggunaan fungsi XFER dalam PLC yang digunakan untuk memindahkan data sebanyak 31 *word* dari satu blok lokasi memori menuju ke lokasi blok memori lainnya. Fungsi berguna apabila data yang akan dipindahkan bersifat mutlak sebanyak *n words*.



Gambar 15 *Flowchart* Proses Pembacaan *Incremental Rotary Encoder*

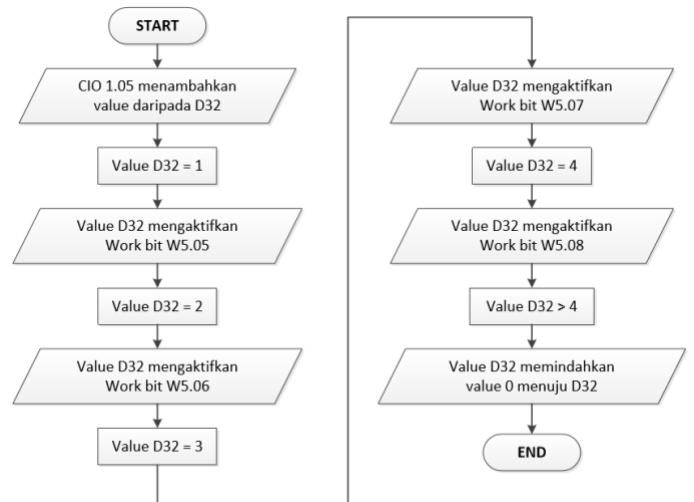
Fungsi program PLC yang digunakan untuk menginput data hasil pembacaan oleh *rotary encoder* menggunakan fungsi PRV. Data kemudian akan disimpan kedalam memori D daripada PLC. Pada gambar 15, merupakan *flowchart* daripada program proses untuk pembacaan 1 *rotary encoder*

seperti, sedangkan pada lengan robot membutuhkan 4 *rotary encoder*.



Gambar 16 *Flowchart* Proses Menjalankan Aktuator Untuk Inialisasi *Home Position Encoder*

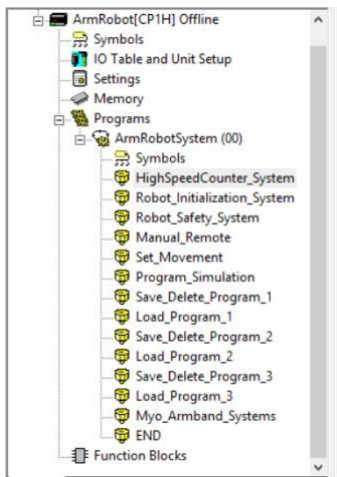
Pada *flowchart* proses menjalankan aktuator untuk inialisasi *home position encoder* dapat dilihat pada gambar 16, digunakan fungsi PULS #0000 - #0003, menandakan bahwa sistem daripada lengan robot menggunakan 4 buah *stepper motor*.



Gambar 17 *Flowchart* *Myo Armband axis selection*

Gambar 17, menjelaskan tentang proses perpindahan dari *axis 1* sampai dengan *axis 4*, karena pada dasarnya gerakan otot lengan operator hanya diterjemahkan dan di proses untuk menggerakkan masing-masing *axis* secara independen.

Program pada *programmable logic control* ini dikelompokkan menjadi beberapa section :



Gambar 18 Pengelompokan program menjadi beberapa *section*

Dapat dilihat pada gambar 18, bahwa:

- *Section 1* = *High speed counter* / program untuk membaca *encoder*
- *Section 2* = Sistem inisialisasi lengan robot
- *Section 3* = Sistem keamanan dari lengan robot
- *Section 4* = Sistem kendali robot manual melalui *touchscreen*
- *Section 5* = Sistem untuk menetapkan koordinat tujuan
- *Section 6* = Simulasi program
- *Section 7* = Sistem untuk menyimpan dan menghapus program 1
- *Section 8* = Sistem untuk memanggil program 1
- *Section 9* = Sistem untuk menyimpan dan menghapus program 2
- *Section 10* = Sistem untuk memanggil program 2
- *Section 11* = Sistem untuk menyimpan dan menghapus program 3
- *Section 12* = Sistem untuk memanggil program 3

III. PENGUJIAN SISTEM

A. Pengujian Pembacaan *Gesture Control*

Pengujian ini dilakukan untuk mengetahui kondisi apa saja yang harus dipenuhi agar data yang diambil saat pembacaan posisi *gesture control* akurat. Berbeda posisi penempatan daripada *gesture control* dapat menjadi *error* dalam sistem. Dari pengujian diperoleh dua cara penempatan *gesture control* pada lengan pengguna.



Gambar 19 Peletakan *gesture control* pada posisi 1



Gambar 20 Peletakan *gesture control* pada posisi 2

Pengambilan *sample* sebanyak sepuluh kali percobaan dan dua hasil penempatan tersebut, pada posisi lengan dimana tekanan otot tidak begitu keras akan diperoleh *error* :

Tabel 1 Tabel tingkat keberhasilan pembacaan *gesture* pada posisi 1

Posisi	<i>Gesture</i>	<i>error</i>	% tingkat keberhasilan
1	Up	1	80 %
	Down	0	100 %
	Left	0	100 %
	Right	0	100 %
	Fist	1	80 %
	Spread	2	60 %
Rata-rata			86.6 %

Tabel 2 Tabel tingkat keberhasilan pembacaan *gesture* pada posisi 2

Posisi	<i>Gesture</i>	<i>error</i>	% tingkat keberhasilan
2	Up	3	40 %
	Down	2	60 %
	Left	3	40 %
	Right	3	40 %
	Fist	4	20 %
	Spread	3	40 %
Rata-rata			40 %

Berdasarkan data percobaan posisi 1 seperti pada gambar 19, dapat dilihat bahwa rata-rata tingkat keberhasilan pembacaan *gesture control* mencapai 86.6 %, *error* disebabkan karena terlalu peka pembacaan sensor *electromyograph*, dan variasi *gesture* yang banyak menyebabkan intervensi antar *gesture*. Sedangkan pada percobaan 2 tingkat keberhasilan pembacaan jauh lebih rendah yaitu di angka 40 %, namun *error* pada posisi 2 seperti pada gambar 20 disebabkan karena tegangan otot tidak terbaca oleh *gesture control*.

B. Pengujian *Logic Converter Output GPIO*

Hasil pengujian *output logic converter*, dimana output yang dihasilkan tidak dapat mencapai persis 24VDC, melainkan 23.40VDC, dimana terjadi selisih sebesar 0.40VDC pada setiap output daripada *logic converter*. Dimana presentase selisih mencapai 1.6 %. Selisih dapat dianggap sebagai toleransi dari penggunaan komponen pada *board converter*.

C. Pengujian Kecepatan Motor Stepper

Dalam pengujian ini akan dilakukan pengamatan perbandingan daripada perhitungan kecepatan motor *stepper* yang seharusnya dengan pengamatan kecepatan motor *stepper*.

Pulse output plc = 1800 hz = 1800 pps (*pulse per second*)

Encoder = 1000 pulsa

$$1 \text{ pulsa} = \frac{360}{1000} = 0.36^\circ$$

$$1800 \text{ pps} = 1800 \times 0.36 = 648^\circ$$

(1 sec \rightarrow 648°)

$$\frac{(648)}{360} \times 1 \text{ sec} = 1.8 \text{ sec}$$

Jika *axis* 1 sedang pada posisi 352° yang tertera pada *display*, maka :

$$\frac{352^\circ \times 1000}{360} = 978^\circ$$

Sedangkan tujuan *axis* 1 berada pada posisi 126°, maka:

$$\frac{126^\circ \times 1000}{360} = 350^\circ$$

Maka selisih kedua koordinat merupakan,

$$978^\circ - 350^\circ = 628^\circ$$

Jika dihitung, maka seharusnya dibutuhkan waktu selama,

$$\frac{628}{648} \times 1.8 \text{ sec} = 1.74 \text{ sec}$$

Motor *stepper* 1.8°, karena dibagi 16 oleh driver maka menjadi,

$$1 \text{ pulsa} = 0.1125^\circ$$

$$1 \text{ revolusi} = 3200 \text{ pulsa}$$

$$1 \text{ revolusi} = \frac{3200}{1800} \times 1 \text{ sec} = 1.77 \text{ sec}$$

Untuk bergerak sebanyak 628°, maka motor *stepper* membutuhkan,

$$\frac{628}{0.1125} = 5.582 \text{ pulsa} = \frac{5.582}{1800} \times 1 \text{ sec} = 3.1 \text{ sec}$$

Pada pengujian secara langsung, diperoleh selisih waktu secara riil dengan waktu perhitungan, selisih mencapai 3.28 sec – 3.1 sec = 0.18 sec = 5.4 %. Dari analisa pengujian didapatkan bahwa faktor selisih dari waktu perhitungan dengan riil adalah murni dari kesalahan instalasi mekanik, karena terjadi slip di saat pertama motor bergerak, sehingga selisih waktu tersebut merupakan selisih waktu ketika motor slip.

IV. KESIMPULAN

Telah dibuat sebuah lengan robot dengan system pemrograman menggunakan metode *kinesthetic* dan *gesture control*. Metode yang saling berkaitan satu dengan yang lain, namun yang membedakan adalah cara penggunaannya. Dari hasil pengujian diperoleh :

- Pemasangan *sling* terhadap *pulley* harus tepat dan tidak kendur, agar tingkat akurasi pergerakan robot sempurna.
- Sensor *electromyograph* membaca pergerakan otot daripada operator robot. Namun peletakan daripada *gesture control* pada lengan pengguna harus mengikuti prosedur yang seharusnya, karena jika tidak dilakukan dengan benar, presentase *error* yang akan dijumpai selama sistem berjalan akan mencapai kurang lebih 60%. Peletakan *gesture control* yang benar juga akan menghasilkan *error rate* sebesar 13.4%.
- *Python* menggunakan program *pyoconnect* yang telah dikembangkan untuk mengolah *raw data* daripada *electromyograph Myo Armband*.
- Digunakan *logic converter* untuk menghubungkan output *raspberry pi* dengan *input PLC* karena *output*

raspberry pi hanya sebesar 3.3V, sedangkan input PLC membutuhkan 24V.

- Pemilihan kualitas komponen dalam membuat *logic converter* dapat menghasilkan *error* pada tegangan yang dihasilkan namun toleransi *error* yang diberikan sebesar 1.6 % dan tidak berpengaruh secara langsung terhadap sistem.

DAFTAR PUSTAKA

- [1] I. Industry, "Teknologi Otomasi untuk Industri yang Lebih Kompetitif," *Technobiz*, 2015. [Online]. Available: <http://indonesianindustry.com/teknologi-otomasi-untuk-industri-yang-lebih-kompetitif/>.
- [2] Tempo.co, "Upah Naik, Bos Kapal Api Pilih Mesin Robot Gantikan Buruh," *Tempo Indonesia 2015*, 2015. [Online]. Available: <https://m.tempo.co/read/news/2015/12/13/090727310/upah-naik-bos-kapal-api-pilih-mesin-robot-gantikan-buruh>. [Accessed: 14-Apr-2016].
- [3] M. Crouse, "Elon Musk Shows Off Tesla Factory's Production Line Robots," *Manufacturing.net*, 2015. [Online]. Available: <http://www.manufacturing.net/news/2015/09/elon-musk-shows-tesla-factorys-production-line-robots>.
- [4] K. Robotics, "KUKA smartPAD," 2016. .
- [5] U. Robots, "UR10 Robot Collaborative Industrial Robot," 2015. .
- [6] Myo, "HOW IT WORKS Connect with Myo," 2016. .
- [7] M. Starr, "Myo armbands used to control prosthetic arm," *CNET Tech Culture*, 2016. [Online]. Available: <http://www.cnet.com/news/myo-ambands-used-to-control-prosthetic-arm/>.