KONTROL GERAK ROBOT MENGGUNAKAN *HAND GESTURE RECOGNITION* BERBASIS *NEURAL NETWORK BACKPROPAGATION*

SKRIPSI

Oleh:
AHMAD HABIBIL MUSTOFA
NIM. 14650080

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2019

KONTROL GERAK ROBOT MENGGUNAKAN *HAND GESTURE RECOGNITION* BERBASIS *NEURAL NETWORK BACKPROPAGATION*

SKRIPSI

Diajukan kepada :
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh:
AHMAD HABIBIL MUSTOFA
NIM. 14650080

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKN0OLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2019

HALAMAN PENGAJUAN

KONTROL GERAK ROBOT MENGGUNAKAN *HAND GESTURE RECOGNITION* BERBASIS *NEURAL NETWORK BACKPROPAGATION*

SKRIPSI

Diajukan kepada :

Fakultas Sains dan Teknologi

Universitas Islam Negeri Maulana Malik Ibrahim Malang

Untuk Memenuhi Salah Satu Persyaratan Dalam

Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh:
AHMAD HABIBIL MUSTOFA
NIM. 14650080

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKN0OLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2019

LEMBAR PERSETUJUAN

KONTROL GERAK ROBOT MENGGUNAKAN *HAND GESTURE RECOGNITION* BERBASIS *NEURAL NERWORK BACKPROPAGATION*

SKRIPSI

Oleh :
AHMAD HABIBIL MUSTOFA
NIM. 14650080

Telah Diperiksa dan Disetujui untuk Diuji
Pada Tanggal : Desember 2019

Dosen Pembimbing I	Dosen Pembimbing II
Yunifa Miftachul Arif, M.T	Roro Inda Melani, M.T., M.Sc
NIP. 19830616 201101 1 004	NIP. 19780925 200501 2 008

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

LEMBAR PENGESAHAN

KONTROL GERAK ROBOT MENGGUNAKAN *HAND GESTURE RECOGNITION* BERBASIS *NEURAL NERWORK BACKPROPAGATION*

SKRIPSI

Oleh :

AHMAD HABIBIL MUSTOFA
NIM. 14650080

Telah Dipertahankan di Depan Dewan Penguji Skripsi
Dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Pada Tanggal : Desember 2019

Susunan Dewan Penguji			Tanda Tangan
1. Penguji Utama	:	Fresy Nugroho, M.T NIP. 19710722 201101 1 001	( )
2. Ketua Penguji	:	Hani Nurhayati, M.T NIP. 19780625 200801 2 006	( )
3. Sekretaris Penguji	:	Yunifa Miftachul Arif, M.T NIP. 19830616 201101 1 004	( )
4. Anggota Penguji	:	Roro Inda Melani, M.T., M.Sc NIP. 19780925 200501 2 008	( )

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini:

Nama : Ahmad Habibil Mustofa

NIM : 14650080

Fakultas/Jurusan : Sains dan Teknologi / Teknik Informatika

Judul Skripsi : **KONTROL GERAK ROBOT MENGGUNAKAN *HAND GESTURE RECOGNITION* BERBASIS *NEURAL NETWORK BACKPROPAGATION***

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 26 Desember 2019

Yang membuat pernyataan

Ahmad Habibil Mustofa

NIM. 14650080

HALAMAN MOTTO

HALAMAN PERSEMBAHAN

الحَمدُ لِلّهِ رَبِّ العَالَمِيْن

Puji syukur kehadirat Allah, shalawat dan salam bagi Rasul-Nya

Saya persembahkan sebuah karya ini kepada:

Kedua orang tua yang amat sangat saya cintai, Abi Muhammad Sabran A. dan Umi Khadija Salila.

Kakak saya yang selalu memberi semangat dan doa, Nur Fajriati, Maulidiatul Hasanah, Roudhotul Jannah dan onyil-onyilnya.

Seluruh dosen Teknik Informatika UIN Maulana Malik Ibrahim Malang, serta seluruh guru yang telah membimbing dan memberikan ilmu kepada saya.

Keluarga Biner (Teknik Informatika angkatan 2014), serta seluruh keluarga besar Teknik Informatika UIN Maulana Malik Ibrahim Malang.

Keluarga Mahasiswa Teladan, yang sering telat datang pulang duluan.

Keluarga B3.

Keluarga ADC.

Dulur Kontrakan Warga 69, Kontrakan K12, dan Kontrakan 5B yang isinya itu itu saja sebenarnya.

Orang-orang yang saya sayangi, yang tak bisa saya sebutkan satu per satu.

Saya ucapkan terimakasih yang luar biasa. Semoga ukhwah kita tetap terjaga dan selalu diridhoi Allah SWT.

Kepada semua pembaca saya mohon doa dan al-fatihah untuk semua kyai, ustadz, guru, dosen, serta pembimbing saya, semoga mereka semua diampuni segala dosa, dimudahkan rezeki dan saya mendapat barokah ilmunya. Allahumma Aamiin.

Karya penelitian ini saya persembahkan kepada umat muslim dan bangsa Indonesia untuk digunakan dan dikembangkan sebaik-baiknya.

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Alhamdulillah sebagai ucapan puji syukur kehadirat Allah SWT. yang telah melimpahkan Rahmat, Taufik serta Hidayah-Nya sehingga penulis mampu menyelesaikan skripsi ini dengan lancar. Sholawat serta salam semoga tetap tercurahkan kepada Nabi Muhammad SAW beserta keluarga, para sahabat, serta pengkikutnya hingga akhir zaman.

Pada penyelesaian skripsi ini banyak sekali pihak yang turut memberikan bantuan secara langsung mau pun tidak langsung. Atas segala bantuan yang telah diberikan penulis ingin menyampaikan terimakasih yang sedalam-dalamnya serta doa kepada :

1. Bapak Yunifa Miftachul Arif, M.T., selaku dosen pembimbing I yang selalu membimbing serta mengarahkan dalam penyusunan skripsi ini.
2. Ibu Roro Inda Melani, M.T., M.Sc., selaku dosen pembimbing II yang selalu menjadi teman dan mentor dalam diskusi keilmuan dan kehidupan.
3. Bapak Dr. Cahyo Crysdian, selaku ketua jurusan Teknik Informatika yang selalu mendukung berbagai penelitian dibidang robotic serta memberikan ilmu baik bidang komputer maupun bidang keilmuan lain yang bermanfaat.
4. Segenap dosen Teknik Informatika yang memberikan bimbingan keilmuan selama studi.
5. Keluarga komunitas ONTAKI dan keluarga mahasiswa Teknik Informatika angkatan 2014 (BINER).
6. Seluruh keluarga besar Teknik Informatika UIN Malang.

Berbagai kekurangan maupun kesalahan mungkin pembaca temukan dalam penulisan skripsi ini. Untuk itu penulis terbuka atas segala kritik dan saran yang membangun dari pembaca sekalian. Adapun hasil test dan cara pemakaian alat pada skripsi ini dapat dilihat di tautan **bit.ly/HasilTestAlat**. Untuk menyambung silaturahmi dan berdiskusi, penulis bisa dihubungi di **ahmad.mustofa71@gmail.com.** Semoga apa yang menjadi kekurangan bisa disempurnakan oleh peneliti selanjunya dan semoga penelitian ini dapat memberikan manfaat. *Wassalamu'alaikum Wr. Wb.*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Mustofa, Ahmad Habibil. 2019. **Kontrol Gerak Robot Menggunakan *Hand Gesture Recognition* Berbasis *Neural Network Backpropagation.*** Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Yunifa Miftachul Arif, M.T., (II) Roro Inda Melani, M.T., M. Sc.

Kata Kunci: *Hand Gesture Recognition, Neural Network Backpropagation, Remote Control, Accelerometer, Gyroscope, GY-521, STM32F01*

Berbeda dengan *remote control* konvensional yang memiliki banyak tombol sehingga menuntut pemahaman baru bagi orang yang belum pernah menggunakannya, kontrol dengan pendekatan *Hand Gesture Recogniton* mampu menawarkan kesederhanaan dan pemahaman instan dengan menggunakan *gesture* tangan pengontrol sendiri.

Kendali gerak robot pada penelitian ini menggunakan *Hand Gesture Recognition* berbentuk *glove*. Sensor MEMS (*Accelerometer* dan *Gyroscope*) dipasang di *glove* tersebut untuk membaca *gesture* tangan. Hasil pembacaan diklasifikasi menggunakaan metode *Neural Network Backpropagation*. Hasil klasifikasi digunakan sebagai perintah kontrol gerak terhadap sebuah robot yaitu berupa perintah diam, maju, mundur, belok kiri dan belok kanan.

Uji coba dilakukan oleh 1 orang *tester*. Uji coba dilakukan sebanyak 10 kali untuk mengenali *gesture* diam, maju, mundur, belok kiri dan belok kanan. Hasil pengenalan seluruh *gesture* sebesar 82,8%.

ABSTRACT

Mustofa, Ahmad Habibil. 2019. **Robot Motion Control Using Hand Gesture Recognition Based On Neural Network Backpropagation.** Undergraduate Thesis. Informatics Engineering Department, Faculty of Science and Technology, Islamic State University of Maulana Malik Ibrahim Malang. Advisor: (I) Yunifa Miftachul Arif, M.T., (II) Roro Inda Melani, M.T., M. Sc.

Keyword: *Hand Gesture Recognition, Neural Network Backpropagation, Remote Control, Accelerometer, Gyroscope, GY-521, STM32F01*

Unlike conventional remote controls that have many buttons that demand a new understanding for people who have never used them, controls with Hand Gesture Recogniton approach are able to offer instant simplicity and understanding by using your own hand gestures.

Control of robot movement in this study using Hand Gesture Recognition in the form of glove. The MEMS Sensor (Accelerometer and Gyroscope) is mounted on the glove to read hand gestures. The reading results are classified using the Neural Network Backpropagation method. The results of the classification are used as a motion control command to a robot in the form of stop, forward, backward, turn left and turn right.

The trial was conducted by 1 person tester. The trial is done 10 times to recognize the gesture of stop, forward, backward, turn left and turn right. The success rate of recognizing of all gestures is 82.8%.

الملخّص

المصطفى, احمد حبيبى. 2019. روبوت مراقبة الحركة باستخدام *Hand Gesture Recognition* استنادا إلى *Neural Network Backpropagation* . أطروحة جامعية. قسم المعلوماتية ,كلية العلوم والتكنولوجيا ,الجامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج.
المشرفة : (I) يونيفا مفتاح العارف، الماجستير. (II) رورو إندي ميلاني، الماجستير.

كلمات البحث : *Neural Network Backpropagation, Hand Gesture Recognition* ، التحكم عن بعد ، التسارع ، الجيروسكوب ، GY-521 ، STM32F01

بخلاف أدوات التحكم عن بُعد التقليدية التي تحتوي على العديد من الأزرار التي تتطلب تفهمًا جديدًا للأشخاص الذين لم يستخدموها أبدًا ، فإن أدوات التحكم باستخدام نهج *Hand Gesture Recognition*قادرة على تقديم البساطة والفهم الفوريين باستخدام إيماءات اليد الخاصة بك.

السيطرة على حركة الروبوت في هذه الدراسة باستخدام التعرف على حركات اليد في شكل قفاز . يتم تثبيت مستشعر *MEMS* (التسارع وجيروسكوب) على القفاز لقراءة إيماءات اليد .يتم تصنيف نتائج القراءة باستخدام طريقة .*Neural Network Backpropagation* تُستخدم نتائج التصنيف كأمر للتحكم في الحركة للروبوت في شكل الإيقاف ، والأمام ، والخلف ، واستدر لليسار ويستدير لليمين.

أجريت التجربة بواسطةوالتق شخص واحد .يتم إجراء التجربة 10 مرات للتعرف على لفتة الإيقاف ، دم ، والرجوع للخلف ، والانعطف يسارًا واليمين .نسبة النجاح في التعرف على جميع الإيماءات هي 82.8 ٪.

BAB I
PENDAHULUAN

1.1. Latar Belakang

Robot kini sudah memasuki hampir seluruh lini kehidupan manusia. Di luar industri, robot telah banyak digunakan di berbagai bidang. Mulai dari manufaktur, rumah, agrikultur, kesehatan, lingkungan, transportasi, komunikasi, *customer service*, edukasi hingga keamanan. Penelitian mengenai perkembangan robot telah banyak dilakukan di berbagai belahan dunia. Seiring perkembangannya yang pesat, konsentrasi penelitian juga merambah ke ranah pengendalian. Bagaimana menciptakan kendali dan kontrol yang lebih akurat, mudah dan dapat digunakan senatural mungkin.

Para peneliti menyadari bahwa gerakan yang terinspirasi oleh bahasa isyarat dapat digunakan sebagai perintah sederhana untuk antarmuka komputer. Hal ini menjadi cikal bakal metodologi *Hand Gesture Recognition* untuk menawaran interaksi manusia dengan komputer yang lebih mudah dan natural (Premaratne, 2014). Oleh karena itu metodologi ini bisa digunakan untuk menjawab tantangan penemuan kendali robot yang lebih akurat, mudah dan dapat digunakan senatural mungkin. Metode ini terus berkembang dengan penggunaan dan pengembangan *accelerometer* yang jauh lebih akurat, penggunaan berbagai macam kamera dan bahkan sensor serat optik yang dapat ditekuk (*optical geniometers*).

Hal ini mengingatkan kita bahwa sesungguhnya manusia adalah makhluk cipataan Allah yang paling sempurna. Banyak sumber ilmu yang bisa dikaji bahkan belum terkuak dari kesempurnaan penciptaan manusia itu sediri. Struktur penyusun hingga cara kerja semua organ pembentuk manusia yang luar bisa kerap dijadikan contoh untuk menyelesaikan permasalahan permasalahan dalam kehidupan sehari-hari. Seperti yang tertulis dalam Al Quran :

﴿لَقَدْ خَلَقْنَا ٱلْإِنسَٰنَ فِىٓ أَحْسَنِ تَقْوِيمٍ ۝٤﴾

Artinya : Sungguh Kami telah menciptakan manusia dalam bentuk dan sifat yang sebaik-baiknya. Dalam tafsir Jalalyn (“Surat At-Tin Ayat 4”, 2019) ayat

ini dijelaskan sebagai berikut : (Sesungguhnya Kami telah menciptakan manusia) artinya semua manusia (dalam bentuk yang sebaik-baiknya) artinya baik bentuk atau pun penampilannya amatlah baik.

Berbeda dengan *remote control* konvensional yang memiliki banyak tombol sehingga menuntut pemahaman baru bagi orang yang belum pernah menggunakannya, kontrol dengan pendekatan ini mampu menawarkan kesederhanaan dan pemahaman instan dengan menggunakan *gesture* tangan pengontrol sendiri. Falcao et al. (2015) menemukan bahwa produk yang berhasil secara teknologi diidentifikasi dengan kecenderungan tidak terlalu tampak "teknis" serta memberikan kemudahan dalam penggunaannya.

Gandhi et al. (2014) mengelompokkan metodologi *Hand Gesture Recognition* kedalam 2 kelompok pendekatan, yakni ; (1) Pendekatan berbasis Visual *(Vision Based Approach)* dan (2) Pendekatan berbasis Sensor (*Sensor Based Approach*). *Vision Based Approach* memanfaatkan data gambar *gesture*. Metode ini berfokus pada gambar yang diambil dari *gesture*, mengekstraksi fitur dan mengenalinya. Masalah yang dihadapi oleh jenis sistem ini adalah kondisi latar belakang yang tidak tetap dan dinamis sehingga mempengaruhi hasil pengenalan. Selain itu sistem ini lebih kompleks dan mahal karena menggunakan perangkat kamera untuk pengenalan dan resource komputasi yang besar untuk memproses gambar *gesture* tersebut. Sedangkan, *Sensor Based Approach* memanfaatkan berbagai sensor untuk mengumpulkan data gerakan yang dilakukan. Data ini kemudian dianalisis dan kesimpulan diambil sesuai dengan model pengenalan. Berbagai sensor ditempatkan di tangan dan ketika tangan melakukan gerakan apa pun, data dicatat dan dianalisis lebih lanjut. *Data glove* adalah contoh teknik berbasis sensor dan ditemukan pertama kai pada tahun 1977 (Dipietro et al., 2008).

Berdasarkan penjelasan tersebut, *Hand Gesture Recognition* dengan *Sensor Based Approach* memiliki potensi yang luas untuk sumbangsih kemajuan kendali robot yang selain murah, tidak terlalu kompleks juga mendekati sifat kendali yang natural sehingga lebih mudah untuk digunakan. Oleh karena itu pada penelitian ini peneliti menawarkan kendali menggunakan *Hand Gesture*

Recognition dengan *Sensor Based Approach* menggunakan sensor *gyroscopescope*. Akan tetapi permasalahan muncul dalam pengolahan nilai sensor *gyroscope* yang digunakan pada metodologi ini. *Gyroscope* MEMS *triple axis*, dapat mengukur rotasi di tiga sumbu, yakni: x, y, dan z. Nilai sumbu akan bersifat positif bila searah putaran jam dan bernilai negatif bila melawan arah putaran jam ("*Gyroscope* - learn.sparkfun.com" t.t.). Nilai ini yang kemudian dimanipulasi untuk mengetahui posisi suatu benda dalam persepsi 3 dimensi. Umumnya posisi ini disebut *yaw*, *pitch* dan *roll*. Ketiga sumbu tersebut mampu menghasilkan nilai positif dan negatif secara bersamaan ketika sensor *gyroscope* sedang bekerja. Hal ini yang mengakibatkan nilai sensor *gyroscope* sangat dinamis pada saat pembacaan secara *realtime*. Oleh karena itu diperlukan metode untuk mengklasifikasi nilai nilai tersebut sesuai kondisi posisi yang diinginkan agar bisa dikenali.

Premaratne (2014) menyimpulkan bahwa metode metode berikut sering digunakan sebagai pengklasifikasi pada *Hand Gesture Recognition*, yakni; *Non Linear SVM, Nearest Neighbor Classification, K-Means Nearest Neighbour Classification, Multi Layer Neural Network*s dan *HMM*. Metode metode tersebut digunakan karena kemampuannya dalam menghadapi masalah klasifikasi non linear yang sering dijumpai di *Hand Gesture Recognition* berbasis *Vision Based Approach*.

Penelitian Zaman Khan (2012) menemukan bahwa *Neural Network* memiliki keberhasilan pengenalan sebesar 98,8% pada *Hand Gesture Recognition* untuk mengenali *American Sign Language* dan *HMM* sebesar 98,3% untuk mengenali 5 gerakan statis dan 12 gerakan dinamis. Dibidang lain Kapsouras et al. (dikutip dari *Engineering Applications of Neural Networks: 14th EANN International Conference, Halkidiki, Greece, September 13-16, 2013 Proceedings, Part I* 2013) menemukan bahwa *neural network* mampu mengidentifikasi jenis tarian daerah yunani berdasarakan video. Martins et al (dikutip dari *Engineering Applications of Neural Networks: 14th EANN International Conference, Halkidiki, Greece, September 13-16, 2013 Proceedings, Part I* 2013) menggunakan *neural network* untuk mengklasifikasi jenis postur duduk

menggunakan kursi yang ditanam beberapa sensor. Metode ini berhasil mengenali 8 postur dengan tingkat keberhasilan sebesar 93,4%.

Dari paparan tersebut, metode *Neural Network* terlihat *Hand*al dalam mengklasifikasi nilai yang dinamis menjadi *output* tertentu. Oleh karena itu, penelitian ini diajukan untuk menggabungkan dua hal tersebut, yakni *Sensor Based Hand Gesture Recognition* dan *Neural Network*. Pada penelitian ini, sensor *gyroscope* MPU6050 digunakan pada metodologi *Sensor Based Hand Gesture Recognition*. Data sensor ini kemudian diolah dengan metode *Neural Network Backpropagation* untuk mengenali jenis gerakan yang akan digunakan sebagai sistem kendali robot. Hasil ini kemudian diuji ke sebuah robot beroda sederhana.

Metode Neural Network ini pada dasaranya juga meruapakan bentuk penerapan kesempurnaan penciptaan manusia seperti yang tertera pada surat At Tin ayat 4 diatas. Sistem kerja otak dalam menerima rangsangan pada neuron, menyimpannya berulang-ulang hingga menjadi pola informasi yang kemudian bisa digunakan oleh manusia untuk menentukan respon dan tindakan ketika menghadapi sesuatu yang kemudian dinamakan “kecerdasan”, juga akhirnya digunakan untuk merekayasa mesin agar mampu berpikir “cerdas” menyerupai manusia dalam prosesnya untuk menyelesaikan masalah. Dari paparan ini jelas terlihat bahwa sungguh manusia diciptakan oleh Allah dalam keadaan yang sebaik - baiknya dan sempurna. Didalamnya terkandung banyak sumber ilmu dan kunci apabila umat muslim mau memperhatikan yang kemudian bisa digunakan untuk menyelesaikan masalah masalah yang ada.

1.2. Rumusan Masalah

Bagaimana mengendalikan robot menggunakan *Motion Based-Hand Gesture Recognition* dengan *Neural Network Backpro* sebagai metode untuk pengklasifikasi *gesture* tangan.

1.3. Tujuan Penelitian

Pengendalian robot menggunakan *Motion Based-Hand Gesture Recognition* dengan *Neural Network Backpro* sebagai metode pengklasifikasi *gesture* tangan.

1.4. Batasan Masalah

- Sampel gerakan tangan pada proses pelatihan *Neural Network Backprogation* berupa *gesture* diam, maju, mundur, kiri dan kanan.
- Menggunakan robot beroda berpenggerak 2 motor .
- Menggunakan sensor *gyroscope* MPU6050.

BAB II
TINJAUAN PUSTAKA

2.1. Penelitian Terkait

Oyedotun dan Khashman (2017) dalam penelitiannya mengusulkan *Hand Gesture Recogniton* menggunakan *deep learning*. *Convolutional Neural Network* (CNN) digunakan untuk mengenali 24 *gesture* tangan *American Sign Language* (ASL). Data gambar ASL tersebut diperoleh dari database publik Thomas Moeslund. Gambar gambar tersebut ditransformasi ke *binary* (black and white) terlebih dahulu untuk segmentasi. Data ini kemudian dijadikan sebagai data latih pada metode CNN dan *autoencoders* (SDAEs). Hasil pengenalan yang dicapai dari penelitian ini mencapai 91.33 % dan 92.83%.

Dalam penelitiannya, Maharani dkk. (2018) mengusulkan *Hand Gesture Recognition* untuk *input* Bioloid Premium Robot menggunakan *K-Means Clustering* dan *Support Vector Machine* (SVM) dengan *Directed Acrylic Graph* (DAG) *Decision*. Sistem ini menggunakan kamera Kinect v2 untuk mengenali *gesture* tangan sebagai *input* maju, kanan, kiri dan berhenti. Percobaan dilakukan oleh 6 orang dengan jarak 2m, 3m dan 4m dengan 3 posisi kemiringan (45, 0 dan -45). Metode SVM membutuhkan waktu 10 detik untuk mengenali *gesture* yang diuji dengan tingkat akurasi 95.15 %. Sedangkan *K-Means* membutuhkan waktu pengenalan selama 4.45 detik dengan tingakt akurasi sebesar 77.42%. Penelitian ini menemukan bahwa *Multiclass* SVM dengan DAG *Decision* lebih baik daripada *K-Means*.

Xu et al. (2012) mengusulkan sistem yang tidak spesifik pada pendekatan ini dengan menggunakan *accelerometer* MEMS. Sistem pengenalan terdiri dari pengumpulan data sensor, segmentasi dan pengenalan. Setelah menerima data percepatan dari perangkat penginderaan, algoritma segmentasi diterapkan untuk menentukan titik awal dan akhir dari setiap isyarat *input* secara otomatis. Urutan tanda isyarat diekstraksi sebagai fitur pengklasifikasian, yaitu yang dijadikan sebagai kode isyarat. Akhirnya, kode gerakan dibandingkan dengan pola standar tersimpan yang telah dipersiapkan sebelumya untuk menentukan gerakan yang paling mungkin.

2.2. Landasan Teori

2.2.1. *Hand Gesture Recognition*

Sejarah *Hand Gesture Recognition* untuk kontrol komputer dimulai dengan penemuan antarmuka kontrol berbasis sarung tangan. Para peneliti menyadari bahwa gerakan yang terinspirasi oleh bahasa isyarat dapat digunakan untuk menawarkan perintah sederhana untuk antarmuka komputer. Penemuan ini berangsur-angsur berkembang dengan pengembangan *accelerometer* yang jauh lebih akurat, kamera inframerah dan bahkan sensor serat optik yang dapat ditekuk (goniometer optik). Beberapa perkembangan dalam sistem berbasis sarung tangan akhirnya menawarkan kemampuan untuk mewujudkan pengenalan berbasis *computer vision* tanpa sensor yang melekat pada sarung tangan. Yakni penggunaan sarung tangan berwarna atau sarung tangan yang mengunakan warna unik (*color marker*) untuk kemampuan pelacakan jari. Selama 25 tahun terakhir, evolusi ini telah menghasilkan banyak produk sukses yang menawarkan koneksi nirkabel total dengan resistensi paling sedikit terhadap pemakainya.

Evolusi sistem isyarat tangan awalnya berdasarkan pada sarung tangan data (*Data glove*). *Data glove* pada intinya adalah antarmuka kabel dengan taktil tertentu atau unit sensorik lainnya yang melekat pada jari atau sambungan sarung tangan, yang dikenakan oleh pengguna. Sakelar taktil, goniometer optik, atau sensor resistansi yang mengukur tekukan sendi menghasilkan pengukuran kasar untuk menentukan *gesture* tangan sedang terbuka atau tertutup dan beberapa sendi jari sedang lurus atau bengkok. Hasil ini dipetakan ke gerakan unik dan ditafsirkan oleh komputer. Keuntungan dari alat sederhana ini adalah tidak ada persyaratan untuk segala jenis pra-pemrosesan. Dengan daya pemrosesan yang sangat terbatas pada komputer pada 1990-an, sistem ini menunjukkan harapan besar meskipun kemampuan manuvernya terbatas karena sambungan yang menghubungkan sarung tangan ke komputer.

Metodologi *Hand Gesture Recognition* secara umum bias dikelompokkan kedalam 2 kelompok pendekatan (Gandhi et al., 2014), yakni ; (1) Pendekatan berbasis Visual *(Vision Based Approach)* dan (2) Pendekatan berbasis Sensor

(*Sensor Based Approach*). *Vision Based Approach* memanfaatkan data gambar *gesture*. Data gambar diambil menggunakan berbagai optic *device* dan kamera. Metode ini berfokus pada gambar yang diambil dari *gesture*, mengekstraksi fitur gambar tersebut dan mengenalinya. *Color marker* adalah salah satu metode pendekatan ini. Terdapat beberapa batasan dalam metode *color marker*. Karena itu gerakan yang dilakukan menggunakan tangan kosong lebih disukai daripada menggunakan *color marker*. Digunakan berbagai algoritma deteksi kulit dalam sistem pengenalan gerakan tangan kosong untuk deteksi tangan. Masalah yang dihadapi oleh jenis sistem ini adalah kondisi latar belakang yang tidak tetap dan dinamis sehingga mempengaruhi hasil pengenalan. Selain itu sistem ini lebih kompleks dan mahal karena menggunakan perangkat kamera untuk pengenalan dan resource komputasi yang besar untuk memproses gambar *gesture* tersebut.

Sensor Based Approach memanfaatkan berbagai sensor untuk mengumpulkan data gerakan yang dilakukan. Data ini kemudian dianalisis dan kesimpulan diambil sesuai dengan model pengenalan. Dalam hal pengenalan isyarat tangan, berbagai sensor ditempatkan di tangan dan ketika tangan melakukan gerakan apa pun, data dicatat dan dianalisis lebih lanjut. Penemuan *Data glove* pertama dilakukan pada tahun 1977 (Dipietro et al., 2008).

2.2.2. Gysroscope

Gyroscope digunakan dalam berbagai aplikasi untuk mengukur laju rotasi pada sumbu yang ditentukan atau mengukur sudut yang diputar. *Gyroscope* gimbal tradisional menggunakan sifat inersia dari roda yang berputar dan hanya dapat mengukur laju rotasi sekitar satu sumbu. Dua atau tiga dari ini harus digunakan jika rotasi sekitar dua atau tiga sumbu ortogonal harus dirasakan. *Gyroscope* bergetar adalah kelas lain dari *gyroscope*, yang menawarkan keunggulan penting dibandingkan desain gimbal. Mereka menawarkan kemungkinan merasakan rotasi sekitar lebih dari satu sumbu, umumnya lebih kecil, membutuhkan daya lebih sedikit dan bantalan dan motor penggerak yang diperlukan dalam desain gimbal dilepas dalam *gyroscope* yang bergetar. Dengan satu pengecualian *gyroscope* bergetar belum dikembangkan untuk kinerja inersia dan penggunaannya terbatas pada tingkat aplikasi belokan (Gallacher et al., 2001).

Gyroscope adalah perangkat untuk mengukur atau mempertahankan orientasi, yang berlandaskan pada prinsip-prinsip momentum sudut. Secara mekanis, *gyroscope* berbentuk seperti sebuah roda berputar atau cakram di mana poros bebas untuk mengambil setiap orientasi. Meskipun orientasi ini tidak tetap, perubahannya dalam menanggapi torsi eksternal jauh lebih sedikit dan berlangsung dalam arah yang berbeda jika dibandingkan dengan tanpa momentum sudut, yang berkaitan dengan tingginya tingkat putaran dan inersia momen. Orientasi perangkat tetap sama, terlepas dari gerak platform pemasangan, karena pemasangan perangkat pada sebuah gimbal akan meminimalkan torsi eksternal.

Cara kerja *gyroscope* yang berlandaskan pada prinsip-prinsip operasi lain juga ada, misalnya *gyroscope* MEMS perangkat elektronik yang ditemukan pada perangkat elektronik konsumen, cincin laser, *gyroscope* optik serat, dan *gyroscope* kuantum yang sangat sensitif.

2.2.3. Quaternion

Cara paling umum untuk merepresentasikan sikap sebuah objek adalah dengan menggunakan seperangkat tiga *euler angle*. Cara ini populer karena mudah dimengerti dan mudah digunakan. Beberapa set *euler angle* begitu banyak digunakan sehingga mereka memiliki nama yang telah menjadi bagian dari bahasa umum, seperti roll, pitch, dan yaw, terutama pada pesawat terbang. Kerugian utama dari *euler angle* adalah: (1) bahwa fungsi-fungsi penting tertentu dari *euler angle* memiliki singularitas, dan (2) bahwa mereka kurang akurat daripada unit angka empat (*quaternion*) ketika digunakan untuk mengintegrasikan perubahan bertahap dalam sikap dari waktu ke waktu. (Diebel, 2006)

Kekurangan dalam representasi *euler angle* ini telah mengarahkan para peneliti untuk menggunakan unit *quaternion* sebagai parameterisasi dari sikap tubuh (orientasi) sebuah objek. Fungsi yang relevan dari unit *quaternion* yang tidak memiliki singularitas dan representasinya yang cocok untuk mengintegrasikan kecepatan sudut tubuh sebuah objek dari waktu ke waktu.

Dibandingkan dengan *quaternion, euler angle* lebih sederhana dan intuitif dan cocok untuk analisis dan kontrol sederhana. Di sisi lain, *euler angle* dibatasi

oleh fenomena yang disebut "*gimbal lock*," yang mencegah mereka mengukur orientasi ketika sudut pitch mendekati +/- 90 derajat. *Quaternion* menyediakan teknik pengukuran alternatif yang tidak mengalami *gimbal lock*. Kuaternon kurang intuitif daripada *euler angle* dan perhitungan matematikanya bisa menjadi sedikit lebih rumit. ("Understanding Quaternions | CH Robotics", 2019)

Dalam matematika, *Quaternion* merupakan perluasan dari bilangan-bilangan kompleks yang tidak komutatif, dan diterapkan dalam mekanika tiga dimensi. *Quaternion* ditemukan oleh ahli matematika dan astronomi Inggris, William Rowan Hamilton, yang memperpanjang aritmetika kompleks nomor ke *quaternion*.

Segera setelah itu penemuan Hamilton, matematikawan Jerman Hermann Grassmann mulai menyelidiki vektor. Meskipun karakter abstrak, fisikawan Amerika JW Gibbs diakui dalam aljabar vektor sistem utilitas besar bagi fisikawan, seperti Hamilton mengakui kegunaan *quaternion*. Pengaruh luas dari pendekatan abstrak yang dipimpin George Boole untuk menulis Hukum Thought (1854), perawatan aljabar dasar logika.

Sebagai himpunan, *quaternion*, berlambang H, sama dengan R4 yang merupakan ruang vektor bilangan riil empat dimensi. H memiliki tiga macam operasi: pertambahan, perkalian skalar dan perkalian *quaternion*. Elemen-elemen *quaternion* ditandakan sebagai $1, i, j$ dan k (i, j dan k adalah komponen imaginer), dan dapat ditulis sebagai kombinasi linear, $a + bi + cj + dk$ (a, b, c, dan d adalah bilangan riil).

Quaternion $p = a + bi + cj + dk$ bisa dituliskan sebagai $p = a + \vec{u}$ di mana $\vec{u}$ adalah vektor 3 bilangan imaginer, $\vec{u} = \{bi + cj + dk\}$. Fungsi rotasi vektor dapat menggunakan operasi *quaternion* daripada operasi matriks riil 4x4, dengan rumus:

$$r = qvq*$$

di mana

$$v = 1 + x_A i + y_A j + z_A k$$

$$q = \cos\frac{a}{2} + \sin\frac{a}{2}x_v i + \sin\frac{a}{2}y_v j + sin\frac{a}{2}z_v k$$

$$r = 1 + x'_A i + y'_A j + z'_A k$$

2.2.4. Arduino

Arduino adalah platform elektronik sumber terbuka yang didasarkan pada perangkat keras dan lunak yang mudah digunakan. Papan Arduino dapat membaca *input* - cahaya pada sensor, jari pada tombol, atau pesan Twitter - dan mengubahnya menjadi *output* - mengaktifkan motor, menyalakan LED, menerbitkan sesuatu secara online. Dengan arduino kita dapat mengatur proses yang berjalan pada arduino dengan memberikan perintah atau fungsi pada baris script untuk mircrocontroler. Bahasa yang digunakan pada pemrograman arduino menggunakan bahasa pemrograman Arduino yang berbasis C (berdasarkan Pengkabelan), dalam Perangkat Lunak Arduino (IDE).

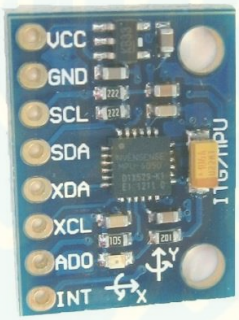
Selama bertahun-tahun Arduino telah menjadi otak ribuan proyek, dari objek sehari-hari hingga instrumen ilmiah yang kompleks. Komunitas pembuat di seluruh dunia - pelajar, penggemar, seniman, programmer, dan profesional - telah berkumpul di sekitar platform open-source ini, kontribusi mereka telah menambah jumlah pengetahuan yang dapat diakses yang luar biasa yang dapat sangat membantu para pemula dan pakar.

Arduino dilahirkan di Ivrea Interaction Design Institute sebagai alat yang mudah untuk membuat prototipe cepat, yang ditujukan untuk siswa tanpa latar belakang dalam bidang elektronik dan pemrograman. Segera setelah mencapai komunitas yang lebih luas, papan Arduino mulai berubah untuk beradaptasi dengan kebutuhan dan tantangan baru, membedakan penawarannya dari papan 8-bit sederhana hingga produk untuk aplikasi IoT, wearable, pencetakan 3D, dan lingkungan tertanam. Semua papan Arduino sepenuhnya open-source, memberdayakan pengguna untuk membangunnya secara mandiri dan akhirnya menyesuaikannya dengan kebutuhan khusus mereka. Perangkat lunak ini juga

bersifat open-source, dan terus berkembang melalui kontribusi pengguna di seluruh dunia (“Arduino - Introduction”, 2019).

2.2.5. Sensor *Gyroscope* MPU 6050

Sensor InvenSense MPU-6050 seperti yang tertera pada Gambar 2.1, berisi *Micro Electro Mechanical Systems* (MEMS) *accelerometer* dan MEMS *gyroscope* dalam satu chip. Sensor ni sangat akurat, karena berisi perangkat keras konversi analog ke digital 16-bit untuk setiap saluran. Oleh karen itu sensor ini menangkap saluran x, y, dan z secara bersamaan. Sensor menggunakan bus-I2C untuk berinteraksi dengan Arduino. MPU-6050 merupakan sensor yang menggabungkan *accelerometer* dan *gyroscope* dengan harga yang terjangkau (“Arduino Playground - MPU-6050”, 2019).



Gambar 2. 1 Sensor *Gyroscope* MPU6050

2.2.6. *Neural Network*

Definisi

Neural Network merupakan salah satu upaya manusia untuk memodelkan cara kerja atau fungsi sistem syaraf manusia dalam melaksanakan tugas tertentu. Pemodelan ini didasari oleh kemampuan otak manusia dalam mengorganisasikan sel-sel penyusunnya yang disebut *neuron*, sehingga mampu melaksanakan tugas tugas tertentu, khususnya pengenalan pola dengan efektivitas yang sangat tinggi. Alexander dan Morton (dikutip dari Suyanto, 2014), mendefinisikan *Neural Network* sebagai prosesor tersebar parallel (*parallel distributed processor*) yang sangat besar yang memiliki kecenderungan untuk menyimpan pengetahuan yang bersifat pengalaman dan membuatnya sipa untuk digunakan. *Neural Network*

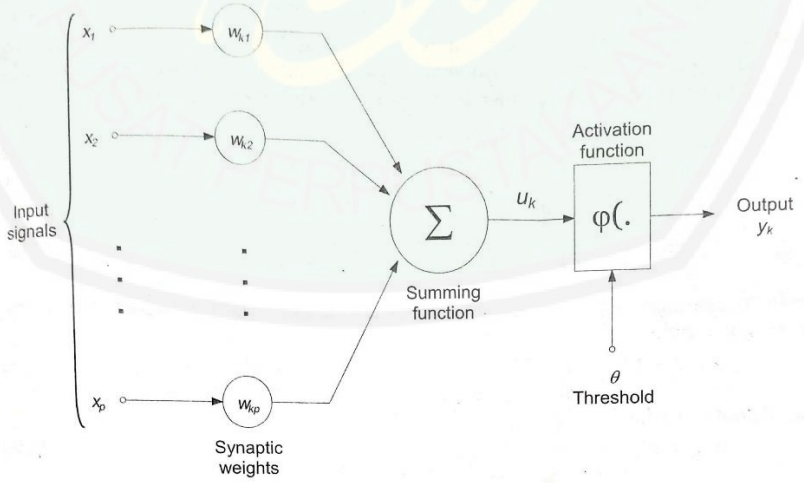
menyerupai otak manusia dalam dua hal, yaitu : Pengetahuan diperoleh jaringan melalui proses belajar; Kekuatan hubungan antar sel syaraf (*neuron*) yang dikenal sebagai bobot-bobot sinaptik digunakan untuk menyimpan pengetahuan.

Model *Artificial Neural Network* (ANN)

Sel sayaraf (*neuron*) adalah unit pemrosesan informasi yang merupakan dasar dari operasi *Neural Network*. Terdapat 3 elemen dasar dari model *neuron*, yaitu :

1. *Synapsis*. Sekumpulan sinapsis atau jalur hubungan, dimana masing masing sinapsis memiliki bobot atau kekuatan jaringan.
2. *Summing Unit*. Suatu *adder* untuk menjumlahkan sinyal sinyal *input* yang diberi bobot oleh sinapsis *neuron* yang sesuai. Operasi operasi yang digambarkan pada pembahasan ini mengikuti aturan *linear combiner*.
3. *Activation Function*. Suatu fungsi aktivasi untuk membatasi amplitudo *output* dari setiap *neuron*.

Menurut Haykin dan Simon (dikutip dari Suyatno, 2014) terdapat tiga variasi model *neuron* yang bisa digunakan karena ketiganya sebenarnya ekivalen. Pertama model *neuron* seperti yang tertera pada Gambar 2.2.



Gambar 2. 2 Model Matematis Nonlinear dari suatu *neuron*

Model ini memasukkan threshold θ_k (ditetapkan secara eksternal) yang memperkecil nilai *input* untuk fungsi aktivasi. Sebaliknya, nilai *input* untuk fungsi aktivasi bisa diperbesar dengan menggunakan bias, yang merupakan kebalikan

dari threshold. Pada Gambar terlihat serangkaian aliran sinyal masukan $x_1, x_2, \dots, x_p$ yang direpresentasikan oleh sebuah *neuron*. Sebuah *neuron* bisa memiliki banyak masukan dan hanya satu keluaran yang bisa menjadi masukan bagi *neuron-neuron* yang lain. Aliran sinyal masukan ini dikalikan dengan suatu penimbang (bobot sinapsis) $wk_1, wk_2, \dots, w_{kp}$ dan kemudian dilakukan penjumlahan terhadap semua masukan yang telah diboboti tadi. Hasil penjumlahan ini disebut keluaran dari the linear combiner μ_k.

Secara matematis sebuah *neuron* k bisa dituliskan dengan pasangan persamaan sebagai berikut :

$$\mu_k = \sum_{j=1}^{p} w_{kj} x_j \tag{2.1}$$

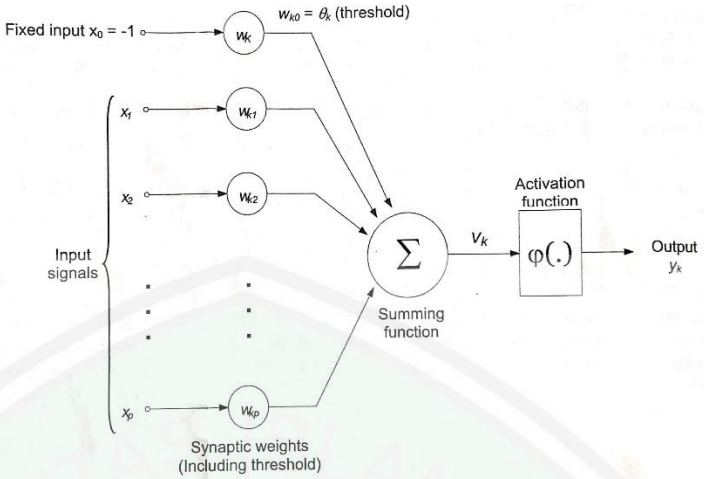
dan

$$y_k = \varphi(\mu - \theta_k) \tag{2.2}$$

Di mana $x_1, x_2, \dots, x_p$ adalah sinyal *input*; $w_{k1}, w_{k2}, \dots, w_{kp}$ adalah bobot – bobot sinaptik dari *neuron* k; μ_k adalah *linear combiner output*; θ_k adalah threshold; $\mu(.)$ adalah fungsi aktifasi; dan y_k adalah sinyal *output* dari *neuron*. Penggunaan threshold memberikan pengaruh adanya affine transformation terhadap *output* μ_k dari linear combiner pada model Gambar 2.2 sebagai berikut:

$$v_k = \mu_k - \theta_k \tag{2.3}$$

Secara khusus, tergantung apakah threshold θ_k adalah positif atau negatif, hubungan antara tingkat aktivitas atau potensial aktivitas internal efektif v_k dari *neuron* k dan linear combiner *output* μ_k dimodifikasi dalam pola seperti pada Gambar 2.3



Gambar 2. 3 Model Matematis Nonlinear dari suatu *neuron* – alternatif 1

Threshold θ_k adalah suatu parameter eksternal dari *neuron* k. Secara ekivalen hal ini bisa dirumuskan kombinasi persamaan (2.4) dan (2.5) sebagai berikut:

$$y_k = \sum_{j=0}^{p} w_{kj} x_j \tag{2.4}$$

dan

$$y_k = \varphi(v_k) \tag{2.5}$$

Pada persamaan (2.4) telah ditambahkan suatu sinapsis baru, yang *input*nya adalah

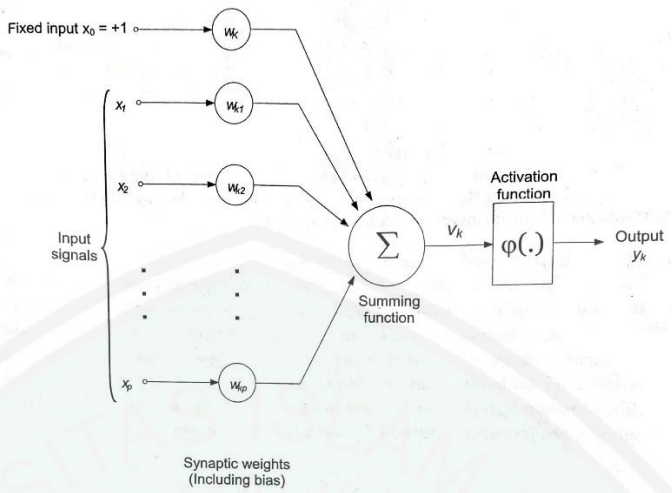
$$x_0 = -1 \tag{2.6}$$

Dan bobotnya adalah

$$w_{k0} = \theta_k \tag{2.7}$$

Oleh karena itu, bisa dirumuskan kembali model *neuron* k seperti Gambar (). Pada gambar tersebut, efek threshold direpresentasikan oleh : penambahan satu sinyal *input* baru yang tetap yaitu -1; dan penambahan bobot sinaptik yang sama dengan threshold θ_k.

Selain itu *neuron* bisa dimodelkan seperti Gambar 2.4



Gambar 2. 4 Model Matematis Nonlinear dari suatu *neuron* – alternatif 2

Dimana kombinasi *input* tetap $x_0 = +1$ dan bobot $w_{k0} = b_k$ berdasarkan bias b_k. Meskipun model *neuron* pada Gambar 2.2, 2.3 dan 2.4 terlihat berbeda, tetapi secara matematis adalah ekivalen.

Fungsi aktifasi

Fungsi aktifasi yang dinotasikan dengan $\varphi(.)$ mendefinisikan nilai *output* dari suatu *neuron* dalam level aktivasi tertentu berdasarkan nilai *output* pengkombinasi linier μ_i. Ada beberapa macam fungsi aktivasi yang biasanya digunakan pada *Neural Network*, diantaranya adalah :

1. *Threshold function*

$$\varphi(v) = \begin{cases} 1 \; if \; v \geq 0 \\ 0 \; if \; v \leq 0 \end{cases} \tag{2.8}$$

2. *Piecewise-linear function*

$$\varphi(v) = \begin{cases} 1 & v \geq \frac{1}{2} \\ v & \frac{1}{2} > v > -\frac{1}{2} \\ 0 & v \leq -\frac{1}{2} \end{cases} \tag{2.9}$$

3. *Sigmoid function*

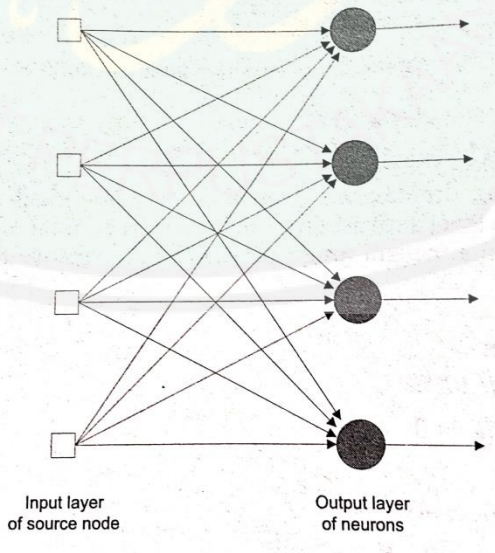
$$\varphi(v) = \frac{1}{1 + \exp(-av)} \tag{2.10}$$

Arsitektur ANN

Pola di mana *neuron-neuron* pada *Neural Network* disusun berhubungan erat dengan algoritma belajar yang digunakan untuk melatih jaringan. Arsitektur jaringan secara umum dibagi menjadi empat, yaitu :

1.1 *Single-Layer Feedforward Networks*

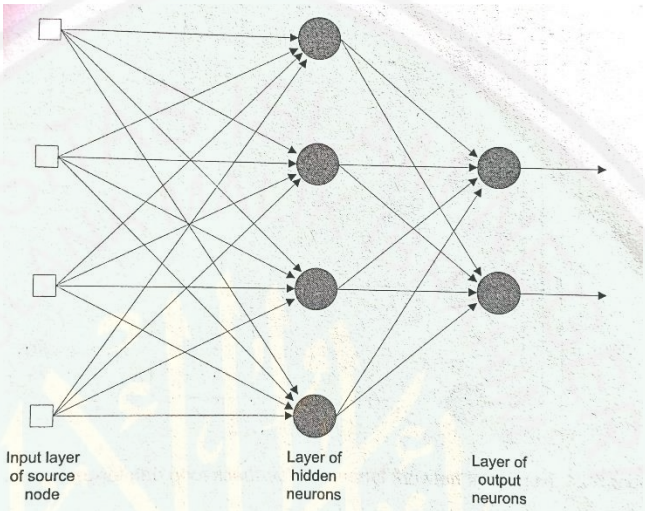
Suatu *Neural Network* berlapis adalah jaringan *neuron* yang diorganisasikan dalam bentuk lapisan-lapisan. Pada bentuk jaringan berlapis yang paling sederhana, hanya terdapat *input layer* dengan node sumber yang terproyeksi ke dalam *output layer* dari *neuron* (*computation nodes*), tetapi tidak sebaliknya. Dengan kata lain, jaringan ini adalah jaringan jenis feedforward yagn tepat. Gambar 2.5 adalah contoh *Neural Network* dengan empat node pada *input layer* dan *output layer*. Jaringan seperti ini disebut *single-layer network*.



Gambar 2. 5 *Feedforward Network* dengan satu lapis *neuron* tunggal

1.2 *Multi-Layer Feedforward Networks*

Kelas ke dua dari feedforward neural network adalah jaringan dengan satu atau lebih lapis tersembunyi (*hidden layer*), dengan *computation nodes* yang berhubungan disebut *hidden neurons* atau *hidden units*. Ilustrasi *multilayer feedforward neural network* untuk kasus satu *hidden layer* seperti yang tertera pada Gambar 2.6

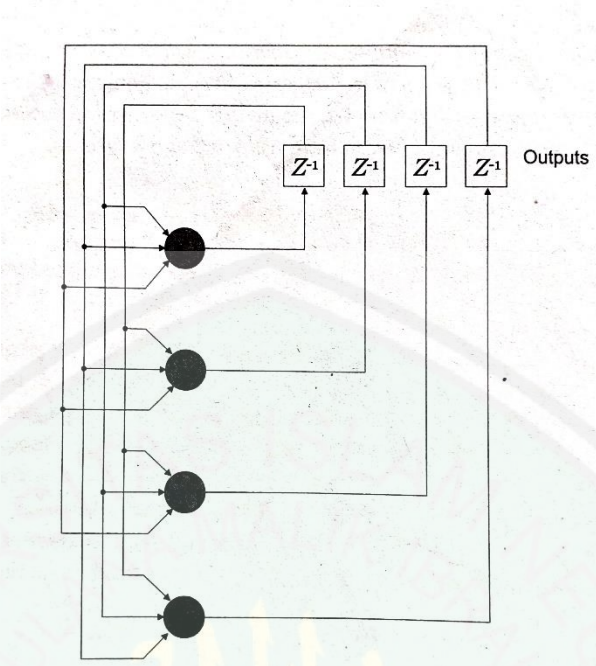


Gambar 2. 6 F*eedforward Network* dengan satu *hidden layer* dan satu *output layer*

Untuk menyingkat jaringan pada Gambar () tersebut, biasanya disebut jaringan 4-4-2, artinya jaringan tersebut memiliki 4 *input neuron*s, 4 *hidden neuron*s, dan 2 *output neuron*s.

1.3 *Recurrent Networks*

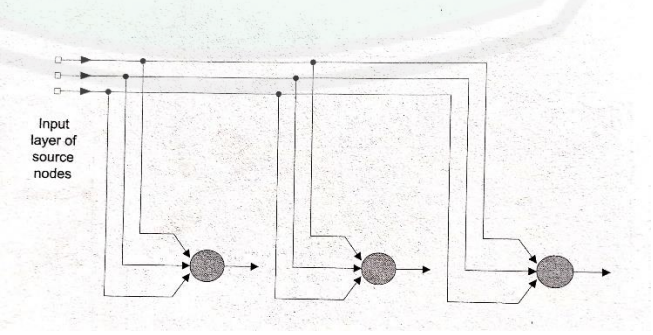
Recurrent neural network adalah jaringan yang mempunyai minimal satu feedback loop. Sebagai contoh, suatu recurrent network bisa terdiri dari satu lapisan *neuron* tunggal dengan masing-masing *neuron* memberikan kembali *output*nya sebagai *input* pada semua *neuron* yang lain, seperti yang diilustrasikan pada Gambar2.7.



Gambar 2. 7 *Recurrent Network* tanpa *self-feedback loop* dan tanpa *hidden neuron*

1.4 *Lattice Structure*

Sebuah lattice (kisi-kisi) terdiri dari satu dimensi, dua dimensi, atau lebih array *neuron* dengan himpunan node sumber yang bersesuaian yang memberikan sinyal *input* ke array; dimensi *lattice* mengacu pada jumlah dimensi ruang dimana graph berada. Graph arsitektur pada Gambar 2.8 menggaambarkan lattice satu dimensi dengan 3 *neuron* yang mendapatkan masukan dari 3 node sumber. Setiap node sumber dihubungkan ke setiap *neuron* dalam lattice.



Gambar 2. 8 *Lattice* satu dimensi dengan 3 *neurons*

Algoritma pembelajaran

Proses belajar pada *Neural Network* adalah proses dimana parameter – parameter bebas *Neural Network* diadaptasikan melalui suatu proses perangsangan berkelanjutan oleh lingkungan dimana jaringan berada. Jenis belajar ditentukan oleh pola dimana pengubahan parameter dilakukan. Sehingga dalam proses belajar terdapat kejadian-kejadian sebagai berikut :

- *Neural Network* dirangasang oleh lingkungan
- *Neural Network* mengubah dirinya sebagai hasil rangsangan ini
- *Neural Network* memberikan respon dengan cara yang baru kepada lingkungan, disebabkan perubahan yang tejadi dalam struktur internalnya sendiri.

Jenis pembelajaran *Neural Network* dapat dibagi menjadi 2 yakni : *Supervised Learning* (Belajar Dengan Pengawasan) dan *Unsupervised Learning* (Belajar Tanpa Pengawasan). Supervised atau active learning adalah proses belajar yang membutuhkan guru. Yakni sesuatu yang memiliki pengetahuan tentang lingkungan. Guru bisa direpresentasikan sebagai sekumpulan sampel *input-output*. Pembangunan pengetahuan dilakukan oleh guru dengan memberikan respon yang diinginkan *Neural Network*. Respon yang diinginkan tersebut merepresentasikan aksi optimum yang dilakukan oleh *Neural Network*. Parameter-parameter jaringan berubah-ubah berdasarkan *vector* latih dan sinyal kesalahan (sinyal kesalahan adalah perbedaan antara keluaran *Neural Network* dan respon yang diinginkan). Proses perubahan ini dilakukan secara berulang-ulang, selangkah demi selangkah, dengan tujuan agar *Neural Network* bisa memiliki kemampuan yang mirip dengan gurunya. *Neural Network* dilatih untuk dapat memetakan sekumpulan sampel *input-output* dengan akurasi yang tinggi.

Unsupervised atau *self-organized learning* tidak membutuhkan guru untuk memantau proses belajar. Dengan kata lain, tidak ada sekumpulan sampel *input-output* atau fungsi tertentu untuk dipelajari oleh jaringan. Contohnya *Neural Network* yang terdiri dari dua lapisan, satu lapis masukan dan satu lapis kompetitif. Lapis masukan menerima data yang disediakan. Lapis kompetitif terdiri dari *neuron-neuron* yang saling bersaing untuk meraih “kesempatan”

memberikan respon ke ciri khas yang berisi data masukan. Dalam bentuk paling sederhana, jaringa beroperasi berdasarkan strategi "*winner-takes-all*".

2.2.7. *Backpropagation*

Backpropagation atau propagasi balik adalah *neural network* yang melakuakn dua tahap perhitungan, yaitu : perhitungan maju untuk menghitung galat antara keluaran actual dan target; dan perhitungan mundur yang mempropagasikan balik galat tersebut untuk memperbaiki bobot-bobot sinaptik pada semua *neuron* yang ada. Berikut langkah-langkah algoritma pelatihan propagasi balik :

1. Definisikan masalah, misal matriks masukan (P) dan matriks target (T).
2. Inisialisasi, menentukan arsitektur jaringan, nilai ambang MSE sebagai kondisi berhenti, *learning rate*, serta menetapkan nilai-nilai bobot sinaptik melalui pembangkitan nilai acak dengan interval nilai sembarang. Pembangkitan nilai acak bisa dalam interval $[-1, +1]$ atau $[-0{,}5, +0{,}5]$ ataupun lainnya. Tidak ada aturan yang baku mengenai hal ini.
3. Pelatihan jaringan
   - Perhitungan Maju

     Dengan menggunakan bobot-bobot yang telah ditentutkan pada inisialisasi awal ($W1$), kita dapat menghitung keluaran dari *hidden layer* menggunakan aktivasi sigmoid berdasarkan persamaan berikut:

     $$A1 = \frac{1}{1 + e^{(W1 * P + B1)}} \tag{2.11}$$

     Hasil keluaran *hidden layer* ($A1$) dipakai untuk mendapatkan keluaran dari *output layer*, seperti pada persamaan berikut:

     $$A2 = W2 * A1 + B2 \tag{2.12}$$

     Keluaran dari jaringan ($A2$) dibandingkan dengan target yang diinginkan. Selisih nilai tersebut adalah *error* (galat) dari jaringan, seperti pada persamaan berikut:

     $$E = T - A2 \tag{2.13}$$

     Sedangkan nilai galat keseluruhan dinyatakan oleh persamaan berikut :

$$SSE = \sum\sum E^2 \quad (2.14)$$

- Perhitungan Mundur

  Nilai galat yang didapat dipakai sebagai parameter dalam pelatihan. Pelatihan akan selesai jika galat yang diperoleh sudah dapat diterima. Galat yang didapat dikembalikan lagi ke lapis-lapis yang berada di depannya. Selanjutnya, *neuron* pada lapis tersebut akan memperbaiki nilai-nilai bobotnya. Perhitungan perbaikan bobot diberikan pada persamaan-persamaan berikut :

  $$D2 = (1 - A2^2) * E \quad (2.15)$$
  $$D1 = (1 - A1^2) * (W2 * D2) \quad (2.16)$$
  $$dW1 = dW1 + (lr * D1 * P) \quad (2.17)$$
  $$dB1 = dB1 + (lr * D1) \quad (2.18)$$
  $$dW2 = dW2 + (lr * D2 * P) \quad (2.19)$$
  $$dB2 = dB2 + (lr * D2) \quad (2.20)$$

- Perbaikan Bobot Jaringan

  Setelah *neuron-neuron* mendapatkan nilai yang sesuai dengan kontribusinya pada galat keluaran, maka bobot-bobot jaringan akan diperbaiki agar galat dapat diperkecil. Perbaikan bobot jaringan diberikan oleh persamaan-persamaan berikut :

  $$TW1 = W1 + dW1 \quad (2.21)$$
  $$TB1 = B1 + dB1 \quad (2.22)$$
  $$TW2 = W2 + dW2 \quad (2.23)$$
  $$TB2 = B2 + dB2 \quad (2.24)$$

- Presentasi Bobot Jaringan

  Bobot –bobot yang baru, hasil perbaikan, dipakai kembali untuk mengetahui apakah bobot-bobot tersebut sudah cukup baik bagi jaringan. Baik bagi jaringan berarti bahwa dengan bobot-bobot tersebut, galat yang akan dihasilkan sudah cukup kecil. Pemakaian nilai bobot-bobot yang baru diperlihatkan pada persamaan-persamaan berikut:

$$TA1 = \frac{1}{1 + e^{(TW1 * P + TB1)}} \quad (2.25)$$

$$TA2 = TW2 * TA1 + TB2 \quad (2.26)$$

$$TE = T - TA2 \quad (2.27)$$

$$TSSE = \sum\sum TE^2 \quad (2.28)$$

Kemudian bobot-bobot sinapsis jaringan diubah menjadi bobot-bobot baru:

$W1 = TW1;$ $B1 = TB1;$ $W2 = TW2;$ $B2 = TB2;$

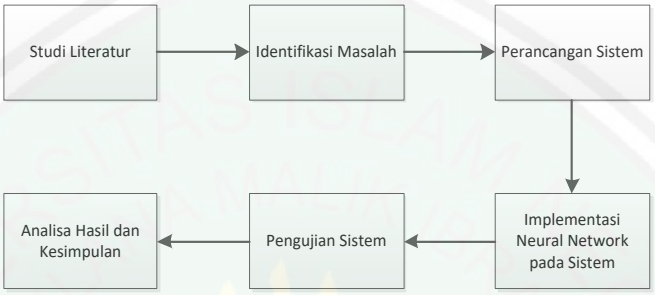
$A1 = TA1;$ $A2 = TA2;$ $E = TE;$ $SSE = TSSE;$

4. Langkah-langkah diatas adalah untuk satu kali siklus pelatihan (satu *epoch*). Biasanaya, pelatihan harus diulang-ulang lagi hingga jumlah siklus tertentu atau telah tercapai SSE (*Sum Square Error*) atau MSE (*Mean Square Error*) yang diinginkan.
5. Hasil akhir dari pelatihan jaringan adalah bobot-bobot W1, W2, B1 dan B2.

BAB III
METODOLOGI PENELITIAN

3.1. Alur Penelitian

Dalam penelitian ini alur yang digunakan seperti yang tertera pada Gambar 3.1.



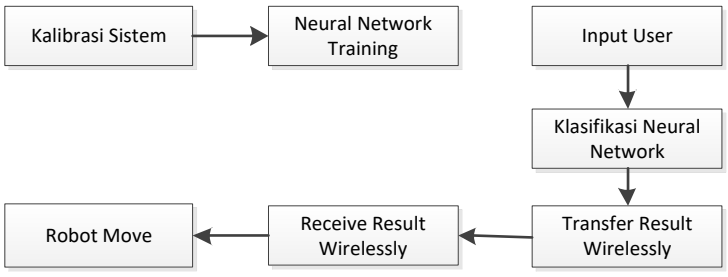
Gambar 3. 1 Alur Penelitian

Studi literatur dilakukan untuk menggali data tentang peneletian penelitian serupa sebelumnya maupun pengertian yang berkaitan dengan penelitian ini. Literatur literatur yang digunakan berasal dari jurnal jurnal internasional, nasional maupun dari penerbit jurnal yang sudah terindeks. Beberapa buku juga digunakan untuk mendukung data data yang digunakan.

Pada perancangan sistem, digunakan STM32f10 sebagai *microcontroller* serta alat komputasi dalam pengimplementasian metode hingga pengujian sistem. IDE Arduino digunakan sebagai *scripting* dan *debugging application.*

3.2. Desain Sistem

Sistem yang bekerja pada penelitian ini digambarkan seperti alur diagram sistem, Gambar 3.2



Gambar 3. 2 Diagram Sistem

3.2.1. Kalibrasi Sistem

Kalibrasi sistem yang digunakan pada sistem ini menggunakan data *Training* yang sudah disediakan sebelumnya. Data tersebut diambil menggunakan sensor *gyroscope* yang diperlakukan sedemikian rupa. Sensor ini nantinya akan digunakan untuk mengambil sampel data *gesture* sebanyak 5 macam, yaitu posisi diam, maju, mundur, kiri dan kanan. Posisi tersebut sesuai yang tertera pada Gambar 3.3.

Gestur Diam | Gestur Maju | Gestur Mundur | Gestur Kiri | Gestur Kanan

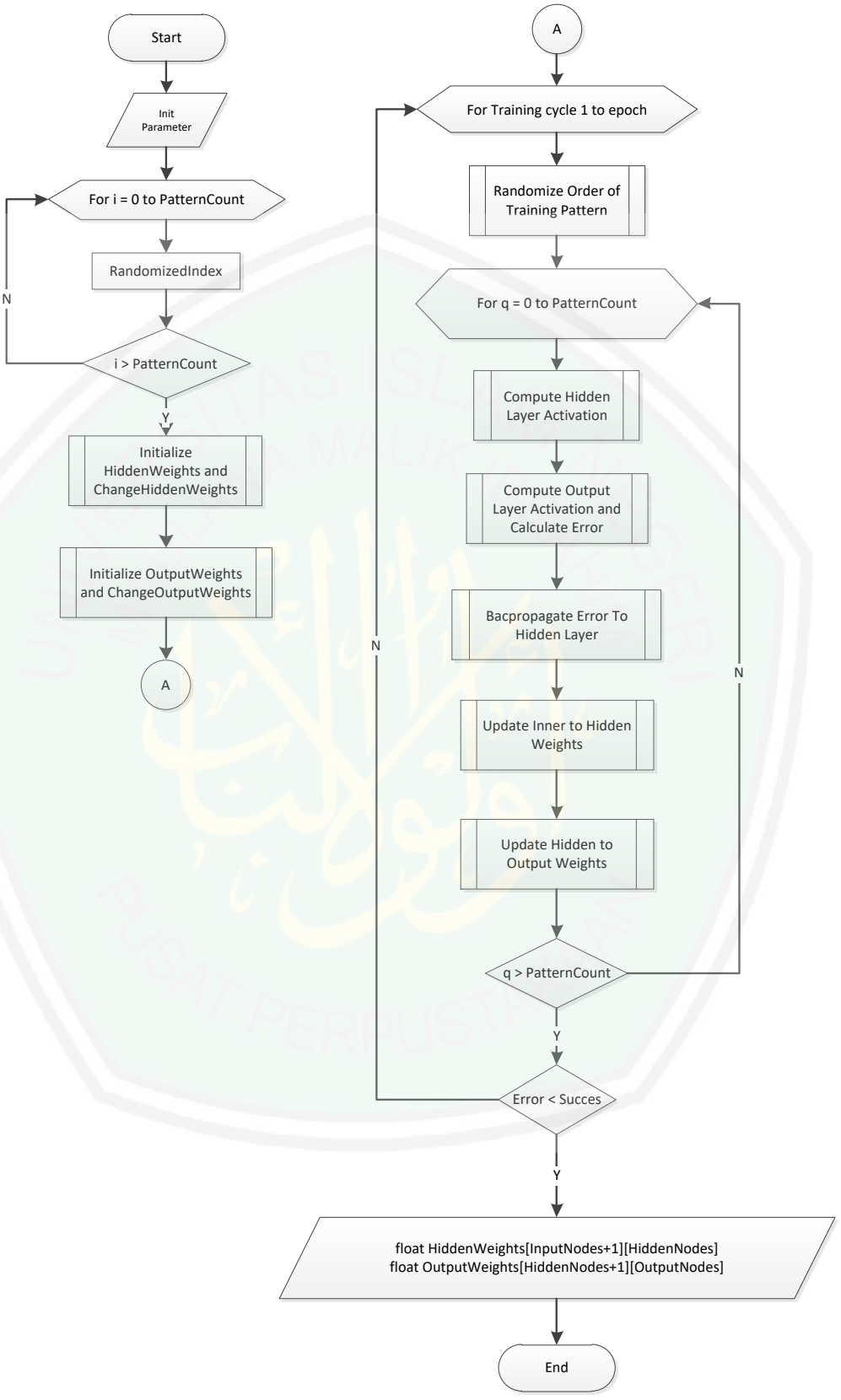
Gambar 3. 3 *Gesture* Tangan

Pada *gesture* diam, pertama sensor diletakan pada *gesture* yang stabil dan datar. Sensor dijalankan selama beberapa detik untuk membaca data *gesture* tersebut. Pada waktu itu sensor membaca sebanyak 500 data dan 100 data terakhir diambil sebagai sampel *gesture* diam. Hal ini dilakukan untuk mengambil nilai yang paling stabil dari sensor pada *gesture* tersebut.

Hal yang sama dilakukan untuk *gesture* maju, mundur, kiri dan kanan. Sensor juga didiamkan selama beberapa detik pada *gesture* tersebut. Sensor membaca sebanyak 500 data dan 100 data terakhir diambil sebagai sampel *gesture* tersebut. Pada akhirnya proses ini menghasilkan data sebanyak 500 buah dengan 100 data pada tiap *gesture* yang nantinya akan digunakan sebagai data latih pada proses NN *Training*. Data ini disimpan pada *SDCard* yang terdapat pada modul *memory card* yang dihubungkan ke *microcontroller*. Hal ini dilakukan guna menghemat penggunaan *memory* EEPROM *microcontroller*.

3.2.2. *Neural Network Training*

Proses *Neural Network Training* yang berjalan pada sistem ini digambarkan seperti *flowchart*, Gambar 3.4



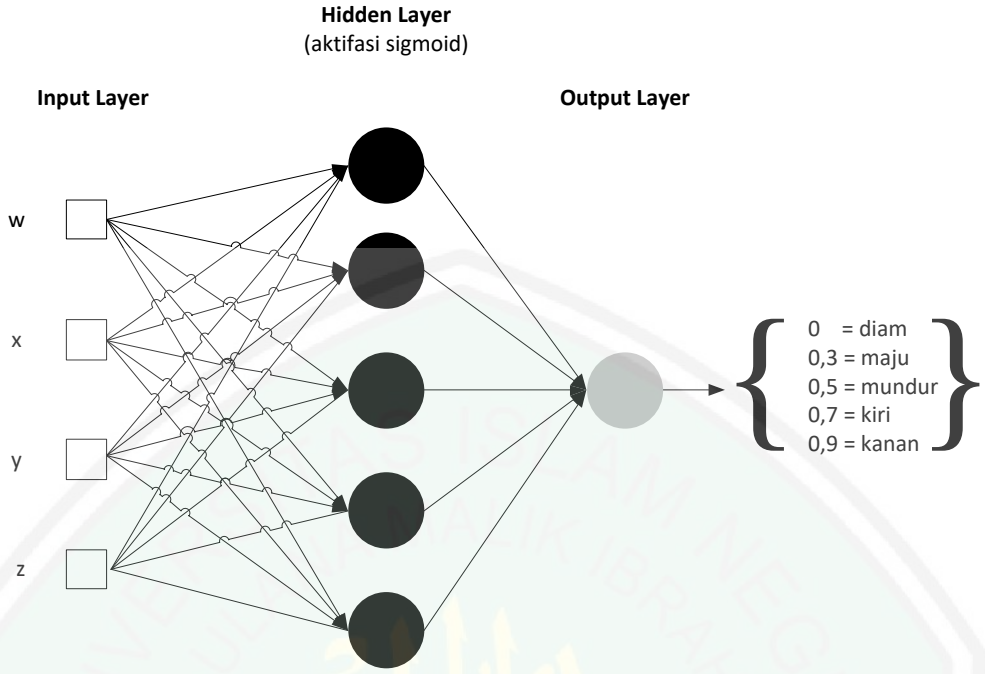
Gambar 3. 4 *Flowchart Neural Network Training*

Pada percobaan sebelumnya dilakukan simulasi arsitektur NN di Matlab untuk mengetahui arsitektur terbaik yang bisa diaplikasikan berdasarkan tingkat akurasi dan pemakaian kebutuhan komputasi yang bisa dilakukan oleh microcontroller yang terbatas. Dari hasil tersebut diketahui bahwa arsitektur terbaik untuk sumber daya tersebut adalah dengan 1 buah *hidden layer* dengan 5 *node* dan 1 buah *output layer* untuk menampung 5 kelas klasifikasi.

Sebelum memasuki proses *Training*, dilakukan inisialisasi parameter parameter yang digunakan. Yaitu jumlah data *gesture* yang digunakan untuk data latih (n), jumlah *input layer* yang digunkan pada proses NN Backpro (x), jumlah *hidden layer* yang digunakan pada NN Backpro (z), jumlah *output layer* yang digunakan pada NN Backpro (y), *Learning Rate* yaitu besar kecepatan proses pelatihan pada proses NN Backpro (η), Momentum (α), nilai bobot awal yang akan diatur pada tiap-tiap *input nodes* (ω) dan batas nilai *error* untuk memutuskan NN sudah dinyatakan berhasil atau belum. Selain itu beberapa variabel *temporary* juga di inisialisasi pada bagian ini untuk membantu proses penghitungan di *Neural Network*.

Pada percobaan sebelumnya dilakukan simulasi arsitektur NN dengan parameter-parameter diatas di Matlab. Dari hasil percobaan didapat bahwa semakin besar nilai Learning Rate maka semakin cepat proses pelatihan namu

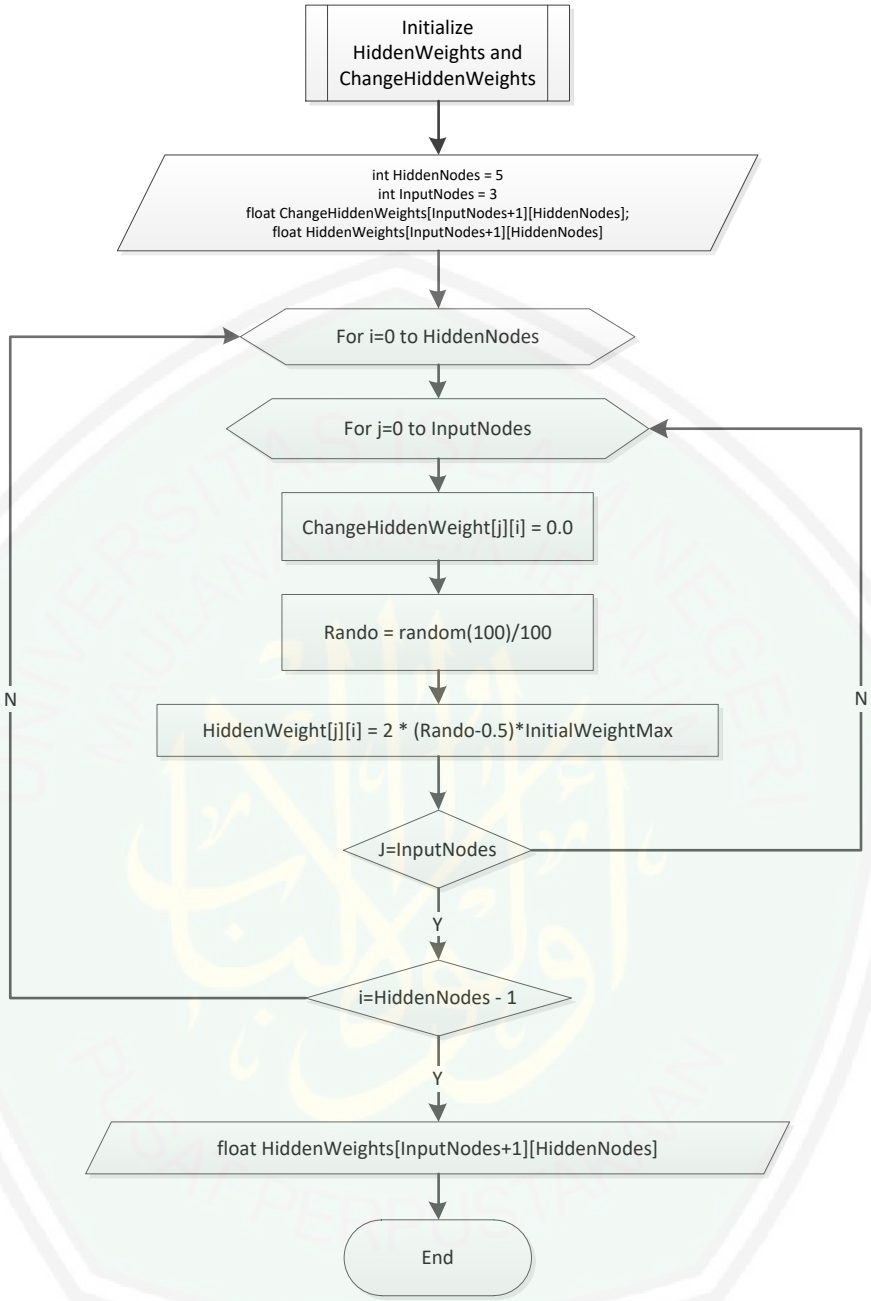
Parameter *Neural Network Backpropagation* yang digunakan pada penelitian ini berupa ; 200 data latih (n), 4 input *layer* (x) yang berasal dari nilai quaternion (w, x, y, z) sensor *gyroscopescope*, 1 *hidden layer* (z) dengan 5 *hidden neuron*, 1 *output layer* (y) dengan 5 kelas klasifikasi, *Learning rate* (η) sebesar 0.01, momentum (α) sebesar 0,9, nilai bobot awal (ω) sebesar 0,5, dan batas nilai *error* sebesar 0,001. Sehingga, Artsitektur NN *Backpropagation* akan tampak seperti ilustrasi pada Gambar 3.5.



Gambar 3. 5 Arsitektur *Neural Network Backpropagation* pada *Gesture Recognition*

- *Initialize and Updating Hidden Weights*

Kemudian proses berlanjut ke tahap Inisialisasi *HiddenWeights* dan *ChangeHidden Weights*. Gambaran alur proses ini seperti yang tertera pada f*lowchart*, Gambar 3.6



Gambar 3. 6 *Flowchart Initialize and Updating Hidden Weights*

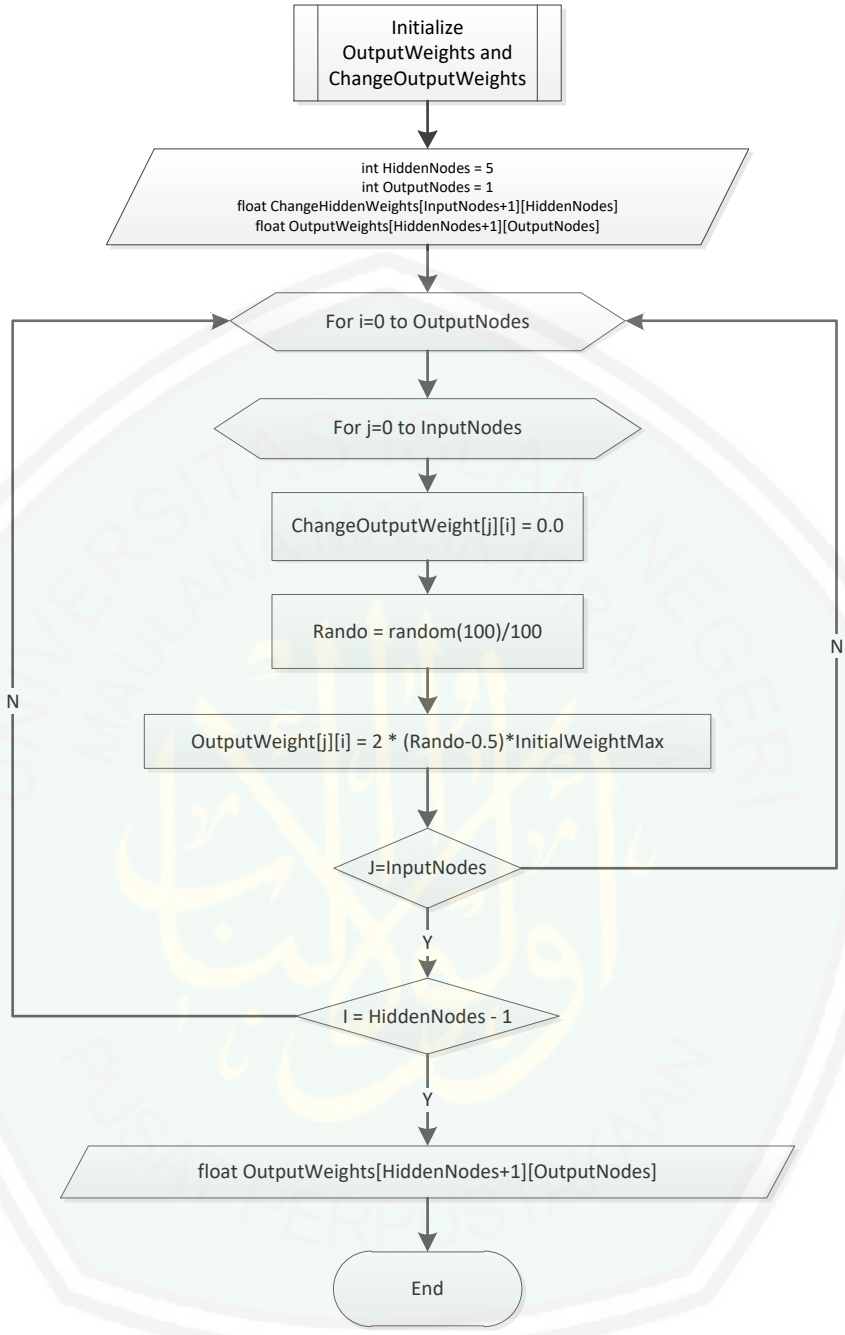
Proses ini mengatur nilai awal *ChangeHiddenWeight* yang nantinya akan digunakan sebagai pengubah nilai *HiddenWeights* ketika perbaikan bobot sinapsis antara *input layer* dan *hidden layer* di *Neural Network*. Pengaturan nilai awal HiddenWeight di seluruh lapisan tersebut menggunakan persamaan (3.2). Hal ini dilakukan untuk mendapatkan nilai bobot awal seminimal mungkin.

$$W_{ij}hidden = 2 \times \left(\frac{rand[0,100]}{100} - 0{,}5\right) \times W_{max} \quad (3.2)$$

InitialWeightMax merupakan *temporary* variabel yang digunakan untuk menampung nilai bobot maksimal pada proses ini.

- *Initialize and Updating Output Weights*

Hal yang sama juga dilakukan untuk bobot antara *hidden layer* dan *output layer*. Dilakukan proses inisialisasi *OutputWeights* dan *ChangeOutputWeights* untuk *layer* tersebut dengan alur proses seperti yang tertera pada *flowchart*, Gambar 3.7



Gambar 3. 7 *Flowchart Initialize and Updating Output Weights*

Proses ini mengatur nilai awal *ChangeOutputWeights* yang nantinya digunakan sebagai pengubah nilai *OutputWeights* ketika perbaikan nilai bobot sinapsis antara *hidden layer* dan *output layer*. Pengaturan nilai awal OutputWeights pada seluruh lapisan tersebut menggunakan persamaan (3.3). Hal ini dilakukan untuk mendapatkan nilai bobot awal seminimal mungkin.

$$W_{ij}output = 2 \times \left(\frac{rand[0,100]}{100} - 0{,}5\right) \times W_{max} \quad (3.3)$$

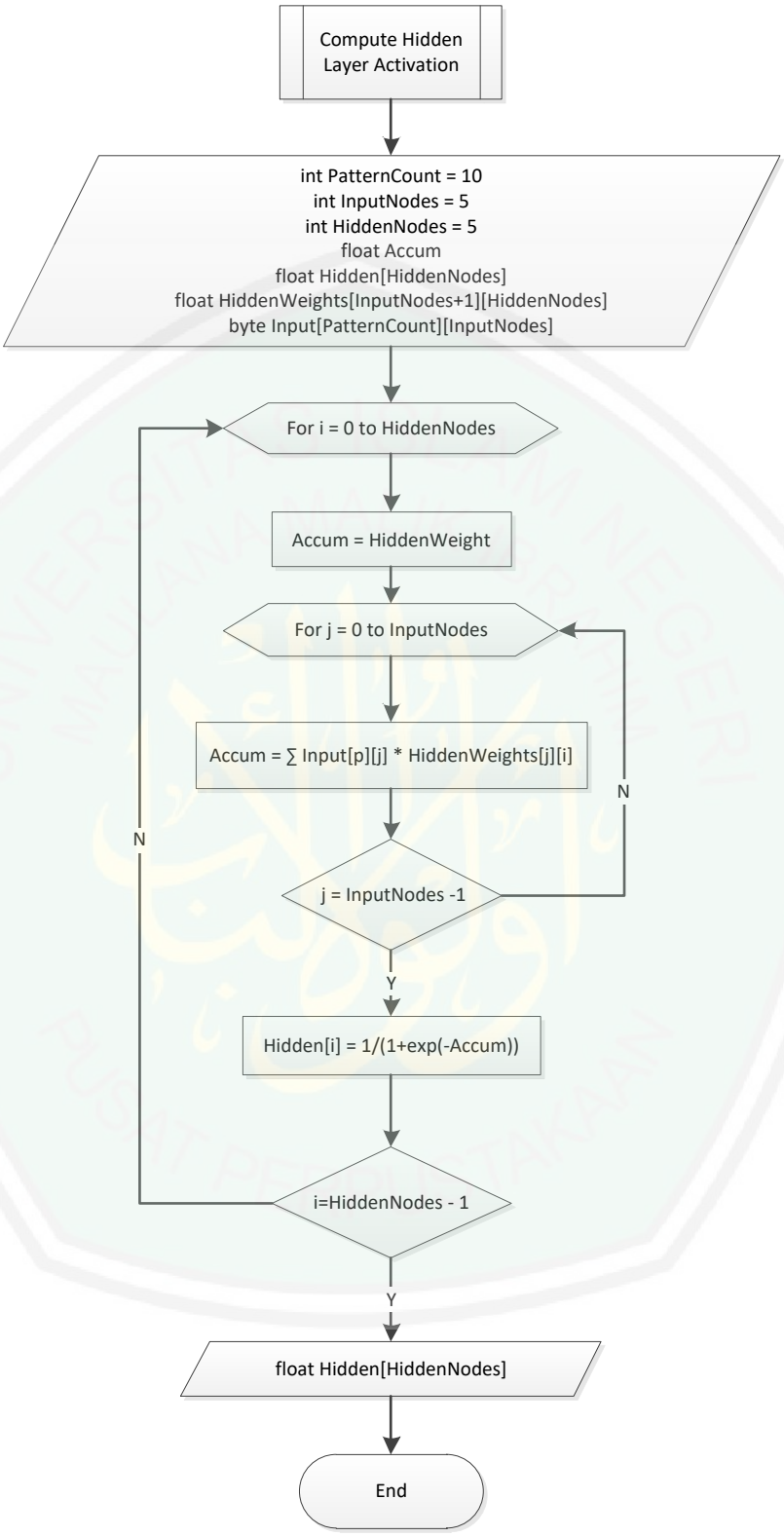
InitialWeightMax merupakan *temporary* variabel yang sama seperti yang digunakan pada proses sebelumnya.

- *Order of Training Pattern*

Selanjutnya dilakukan pengacakan urutan pelatihan pada setiap iterasi untuk mengurangi osilasi dan konvergensi pada *local minimum*.

- *Compute Hidden Layer Activation*

Proses selanjutnya adalah aktifasi nilai *input* berbobot pada *hidden layer*. Alur kerja proses ini seperti yang tertera pada *flowchart*, Gambar 3.8



Gambar 3. 8 *Flowchart Compute Hidden Layer Activation*

Variabel *Accum* merupakan variabel *temporary* yang digunakan untuk menyimpan nilai total dari sinapsis *input* berbobot ke *hidden layer*. Selanjutnya

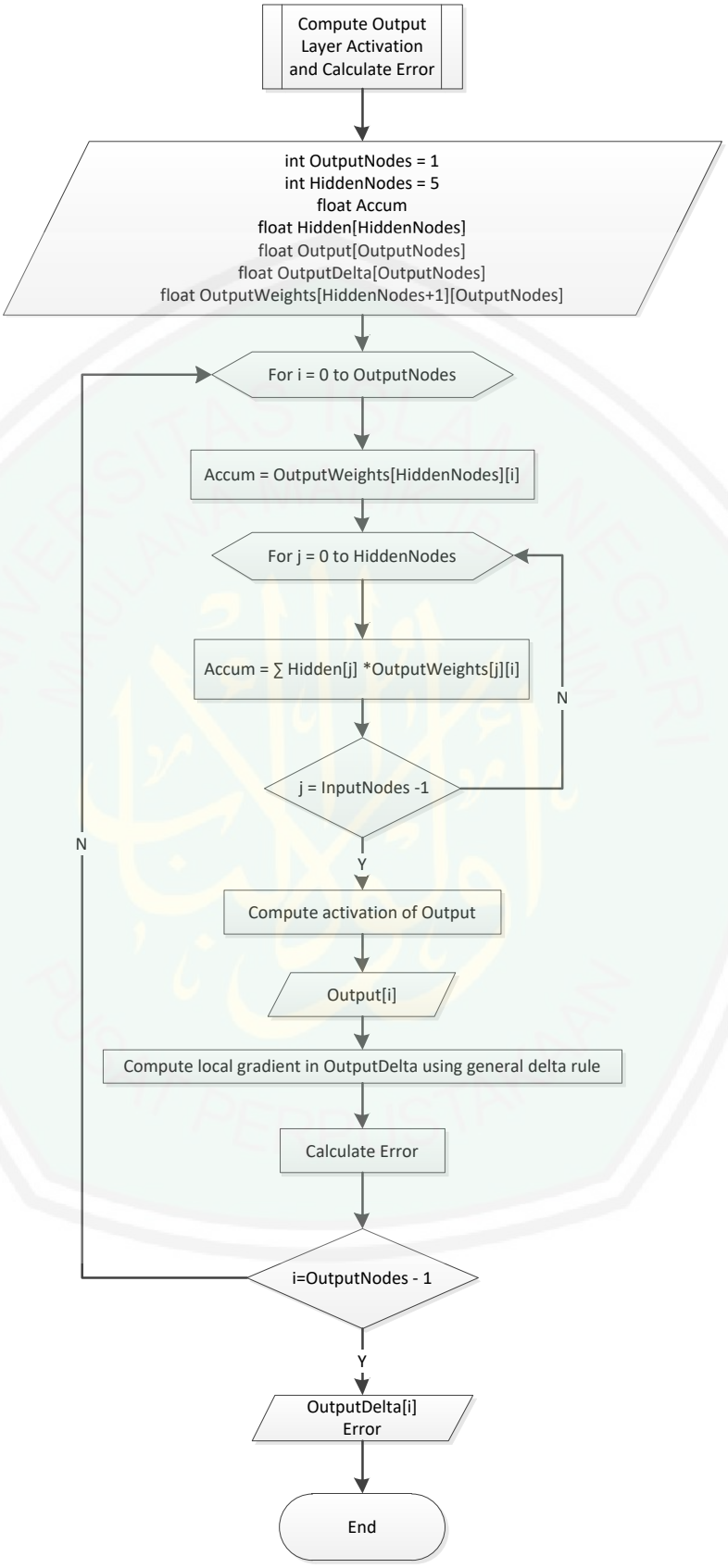
nilai ini diaktifasi menggunakan aktifasi sigmoid agar nilai *output* berkisar antara 0 sampai 1 seperti pada persamaan (3.4).

$$Hidden_i = \frac{1}{1 + e^{-Accum_i}} \quad (3.4)$$

Proses ini dilakukan diseluruh lapisan pada *layer* tersebut.

- *Compute Output Layer Activation and Calculate Error*

Setelah itu hasil aktifasi di *hidden layer* diaktifasi di *output layer* setelah ditambah bobot pada *output layer*. Alur kerja proses ini seperti yang tertera pada *flowchart*, Gambar 3.9



Gambar 3. 9 *Flowchart Output Layer Activation and Calculate Error*

Seperti pada proses sebelumnya variabel *temporary Accum* digunakan untuk menampung nilai total dari sinapsis *hidden* ke *output layer* yang telah diboboti. Nilai ini diaktifasi dengan menggunakan aktifasi sigmoid seperti pada persamaan (3.5)

$$Output_i = \frac{1}{1 + e^{-Accum_i}} \tag{3.5}$$

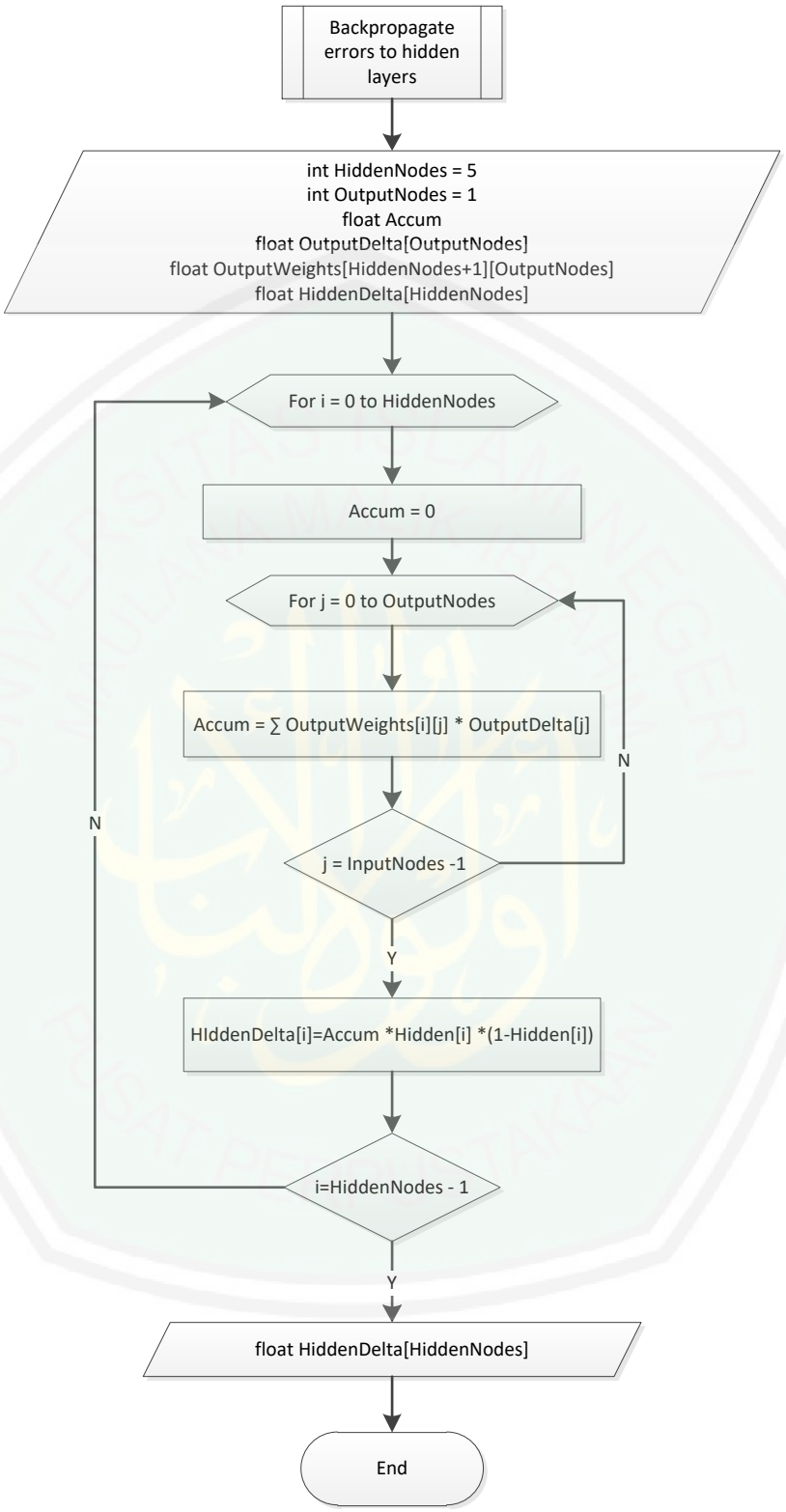
Kemudian dilakukan proses penghitungan *error* antara nilai aktifasi pada *output layer* dengan nilai target sesungguhnya dengan menggunakan delta rule seperti pada persamaan (3.6)

$$OutputDelta_i = (t_i - O_i)O_i(1 - O_i) \tag{3.6}$$

Nilai ini berfungsi sebagai *local gradient decent*. Delta merepresentasikan besarnya kesalahan. Yakni semakin besar delta maka semakin besar perbedaan antara nilai target suatu *neuron* dengan *output* aktualnya sehingga memungkinkan kita untuk menghitung besarnya kesalahan pada setiap *neuron* keluaran, menentukan seberapa banyak setiap koneksi ke *neuron* berkontribusi terhadap kesalahan, dan membuat penyesuaian secara bertahap dan meningkat pada bobot tersebut yang secara sistematis mengurangi kesalahan. Nilai ini disimpan pada variabel OutputDelta . Proses ini dilakukan diseluruh lapisan pada *layer* tersebut.

- *Backpropagate Error to Hidden Layer*

Proses selanjutnya yaitu mempropagasi kembali nilai *error* yang didapat dari proses diatas ke lapisan tersembunyi. Alur kerja proses ini seperti yang tertera pada *flowchart*, Gambar 3.10



Gambar 3. 10 *Flowchart Bacpropagate Error to Hidden Layers*

Pada proses ini dilakukan penjumlahan nilai total *output* berbobot dengan *gradient decent* pada variabel OutputDelta. Nilai ini disimpan pada variabel

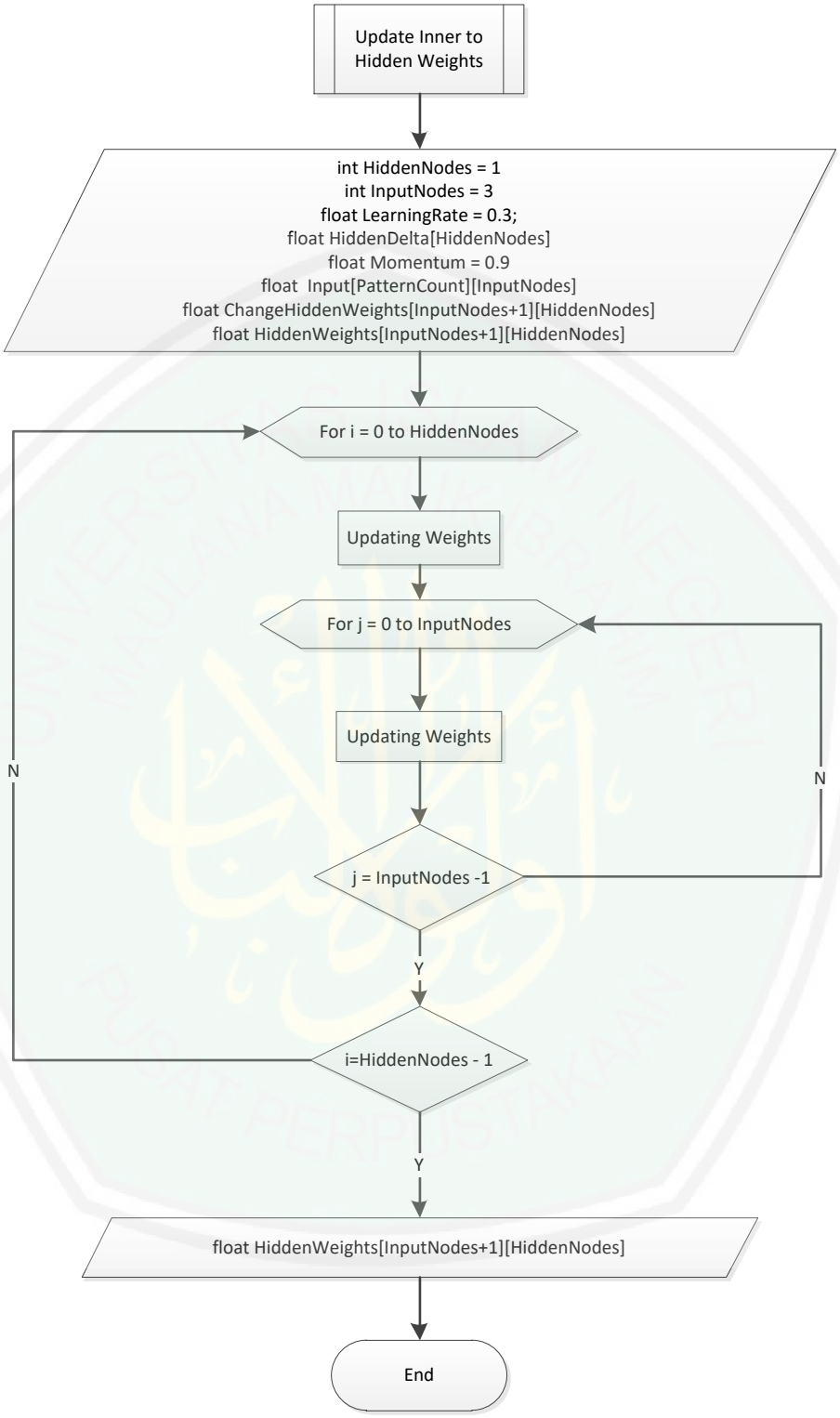
temporary $Accum$. Nilai ini kemudian digunakan untuk menghitung *gradient decent* pada *hidden layer*. Seluruh proses dikerjakan seperti persamaan (3.7)pr

$$HiddenDelta_i = \Sigma_{j=1}^{N_l-1} \delta_j w_{ij}(1 - O_i) \quad (3.7)$$

Nilai tersebut kemudian disimpan pada variabel HiddenDelta. Proses ini dikerjakan diseluruh lapisan pada *layer* ini.

- *Update Inner to Hidden Weights*

Seteleh *gradient decent* pada *hidden* dan *output layer* diketahui, proses selanjutnya yakni memperbarui bobot. Proses pertama yang dilakukan adalah memperbarui bobot pada sinapsis *neuron input* ke *hidden layer*. Alur kerja proses ini seperti yang tertera pada *flowchart*, Gambar 3.11



Gambar 3. 11 *Flowchart Update Inner to Hidden Weights*

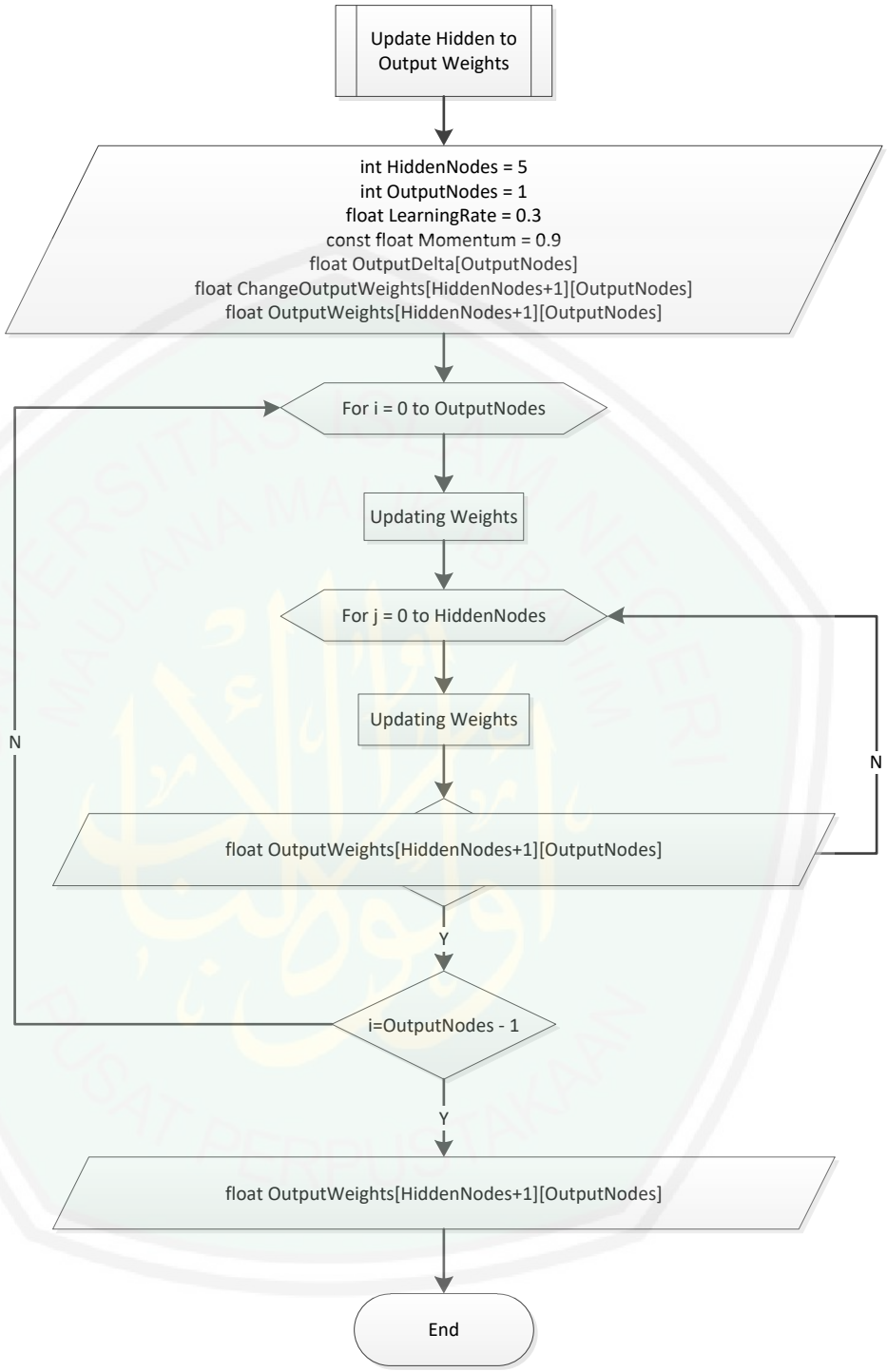
Pada proses ini dilakukan perubahan bobot pada *layer* ini dengan menggunakan persamaan (3.8)

$$HiddenWeight_{ji}^{n} = l_r \delta_i X_{ji} + \alpha \Delta w_{ji}^{(n-1)} \quad (3.8)$$

Dengan l_r sebagai *learning rate*, δ_i sebagai nilai *gradient decent* pada variabel HiddenDelta, X_{ij} sebagai bobot dari *layer*, α sebagai momentum dan Δw_{ji} sebagai selisih bobot awal dengan pengubah.

- *Update Hidden to Output Weights*

Selanjutnya dilakukan proses pembaharuan bobot pada sinapsis *neuron hidden* ke *output layer*. Alur kerja proses ini seperti yang tertera pada *flowchart*, Gambar 3.12



Gambar 3. 12 *Flowchart Update Hidden to Output Weights*

Sama halnya pada proses sebelumnya, pada proses ini dilakukan perubahan bobot pada *layer* ini dengan menggunakan persamaan (3.8)

$$OutputWeight_{ji}^{n} = l_r \delta_i X_{ji} + \alpha \Delta \mathrm{w}_{ji}^{(n-1)} \qquad (3.8)$$

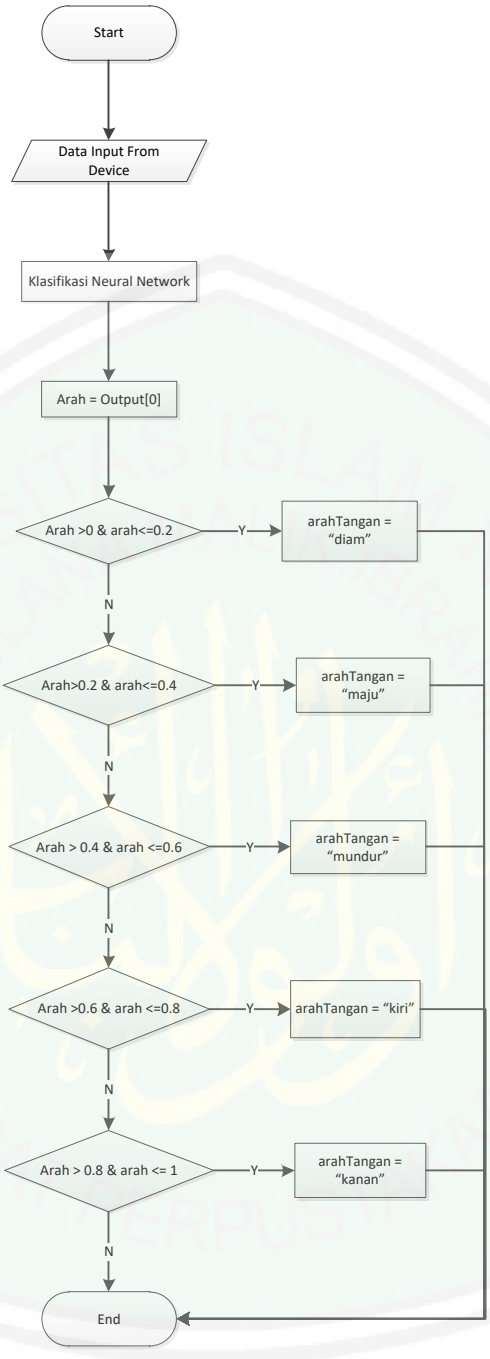
Dengan l_r sebagai *learning rate*, δ_i sebagai nilai *gradient decent* pada variabel OutputDelta, X_{ij} sebagai bobot dari *layer*, α sebagai momentum dan Δw_{ji} sebagai selisih bobot awal dengan pengubah. Proses *Compute Hidden Layer Activation* hingga *Update Hidden Layer to Output Weights* dilakukan sebanyak jumlah data latih. Pada akhirnya, seluruh proses pelatihan dihentikan dan dianggap berhasil jika nilai *error* lebih kecil dari nilai *success* yang telah ditentukan.

3.2.3. *Input User*

Pada proses ini *user* menggunakan *device* kontrol robot pada tangannya. Device ini terdiri dari microcontroller STM32f10 dengan modul sensor *gyroscope* MPU6050 serta modul *radio frequency* nRF24L01. *User* kemudian memeragakan *gesture* arah kendali untuk mengendalikan robot seperti *gesture* yang dipakai pada data latih sebelumnya, yakni *gesture* diam, maju, mundur, kiri dan kanan. Data *gesture* ini kemudian ditangkap oleh sensor *gyroscope* MPU6050. Data ini nantinya akan diklasifikasi menggunakan metode *Neural Network Backpropagation* pada fungsi *testing*. Proses ini akan dijelaskan lebih dalam pada pembahasan selajutnya.

3.2.4. Klasifikasi *Neural Network*

Proses selanjutnya yaitu Klasifikasi *Neural Network*. Proses ini mengklasifikiasi *input user* pada proses sebelumnya menjadi kelas *gesture* gerakan agar bisa dikenali. Alur kerja pada proses ini seperti yang tertera pada *flowchart*, Gambar 3.13



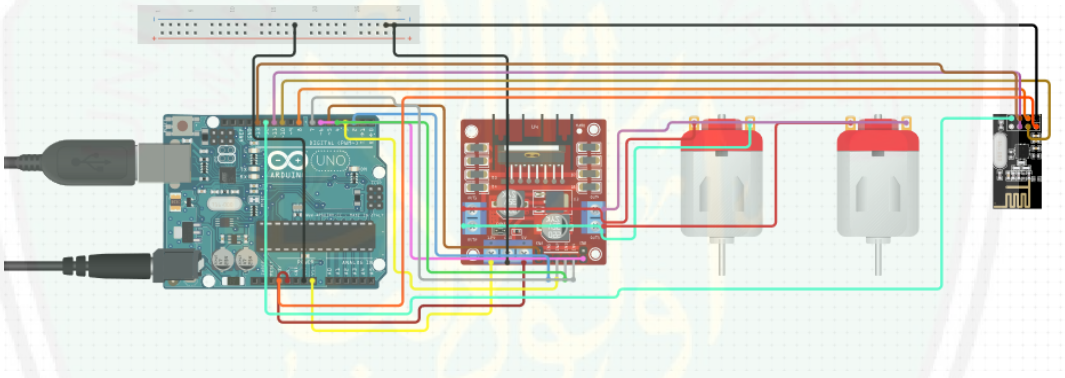
Gambar 3. 13 *Flowchart* Klasifikasi *Neural Network*

Data *input* yang sudah didapat dari proses sebelumnya diklasifikasi menggunakan metode NN Backpro. Pada proses klasifikasi ini data dihitung menggunakan arsitektur NN Backrpo yang sudah terboboti melalui proses *Neural Network Training* pada sub bab 3.2.2. sebelumnya. Seperti pada proses *Neural Network Training* pada Sub bab 3.2.2, data masuk melewati *layer* Input, melewati sinapsis *neuron hidden* berbobot, diaktifasi pada *layer hidden* menggunakan

aktifasi sigmoid dengan persamaan (3.4) seperti pada alur *flowchart* pada Gambar 3.6. Kemudian melewati sinapsis *neuron output* berbobot dan dilanjutkan dengan aktifasi pada *layer* Output mengunakan persamaan (3.5) seperti pada alur *flowchart* pada Gambar 3.7 hingga menghasilkan Output nilai antara 0 hingga 1 sesuai kelas Output yang ditentukan pada proses *Training* pada sub bab 3.2.2. Nilai *output* kemudian dipilah mengunakan *if else* seperti yang tertera pada *flowchart* Gambar 3.13.

3.2.5. *Transfer Result via Radio Frequency*

Pada proses ini hasil *gesture* dari klasifikasi pada proses sebelumnya dikirim dari *device* kontrol ke robot. Arsitektur sistem yang digunakan pada proses ini seperti yang tertera pada Gambar 3.14



Gambar 3. 14 Arsitektur Sistem *Transfer Result via Radio Frequency*

Arsitektur sistem ini menggunakan microcontroller Arduino Uno dan RF module nRF24L01. Pada Gambar 3.12, Pin VCC *radio frequency* dihubungkan pada pin 5V STM32f10. Pin GND *radio frequency* dihubungkan pada pin GND STM32f10. Pin TX *Radio frequency* dihubungkan dengan Pin 19 STM32f10 dan Pin RX *radio frequency* dihubungkan pada pin 18 STM32f10.

3.2.6. *Receive result via Radio Frequency*

Berbeda dari proses sebelumnya yang berlangsung pada *device* glove, proses ini berjalan pada *device* robot. Sistem robot menerima hasil data dari proses sebelumnya pada sub bab 3.2.5. yaitu pengiriman data via *radio frequency* berupa sinyal kendali diam, maju, mundur, kiri dan kanan. Sinyal tersebut berupa bilaangan bulat yang merupakan id per gerakan.

3.2.7. *Robot Move*

Pada proses ini, id yang diterima pada proses 3.2.6 diubah oleh program robot menjadi fungsi gerak untuk driver motor. Program menerjemahkan id gerakan menjadi sebuah fungsi yang menghasilkan nilai pwm untuk mengatur kecepatan roda sehingga robot dapat bergerak maju, mundur, belok kiri dan kanan.

3.3.Rencana Uji Coba

Pengujian Sistem dilakukan oleh 1 orang tester dengan cara memasukkan data uji sebanyak 50 kali pada tiap *gesture* tangan. Data ini kemudian dianalisa dan dinilai tingkat keakuratannya menggunakan MSE.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(f_i - y_i)^2 \quad (3.9)$$

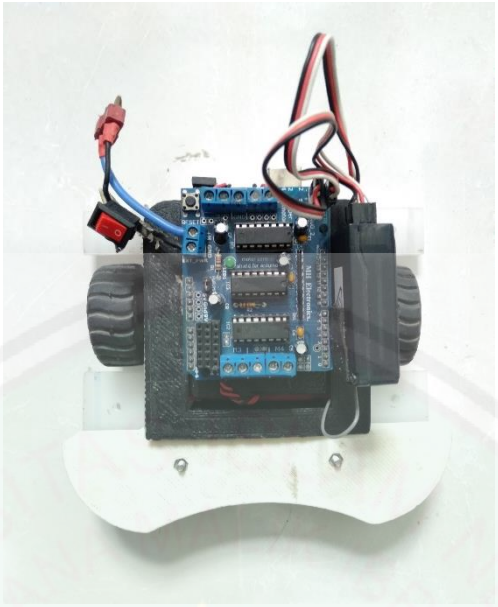
Tingkat keakuratan MSE merupakan rata – rata dari jumlah kuadratik hasil peramalan (f_i) dikurangi hasil sebenarnya (y_i) sesuai dengan persamaan (3.9)

BAB IV
HASIL DAN PEMBAHASAN

4.1. Implementasi *Hardware*

Gambar 4. 1 *Remote control*

Gambar 4.1 merupakan hasil implementasi hardware *remote control* yang terdiri dari Microcontroller STM32F1xx yang berfungsi sebagai pusat komputasi. Modul Sensor IMU 6 Dof MPU6050 yang berfungsi sebagai pembaca nilai gerakan tangan. Modul SDCard yang berfungsi sebagai media penyimpanan data. Modul Wireless NRF24L01 yang berfungsi sebagai alat transmisi hasil klasifikasi yang digunakan sebagai perintah penggerak robot



Gambar 4. 2 Robot

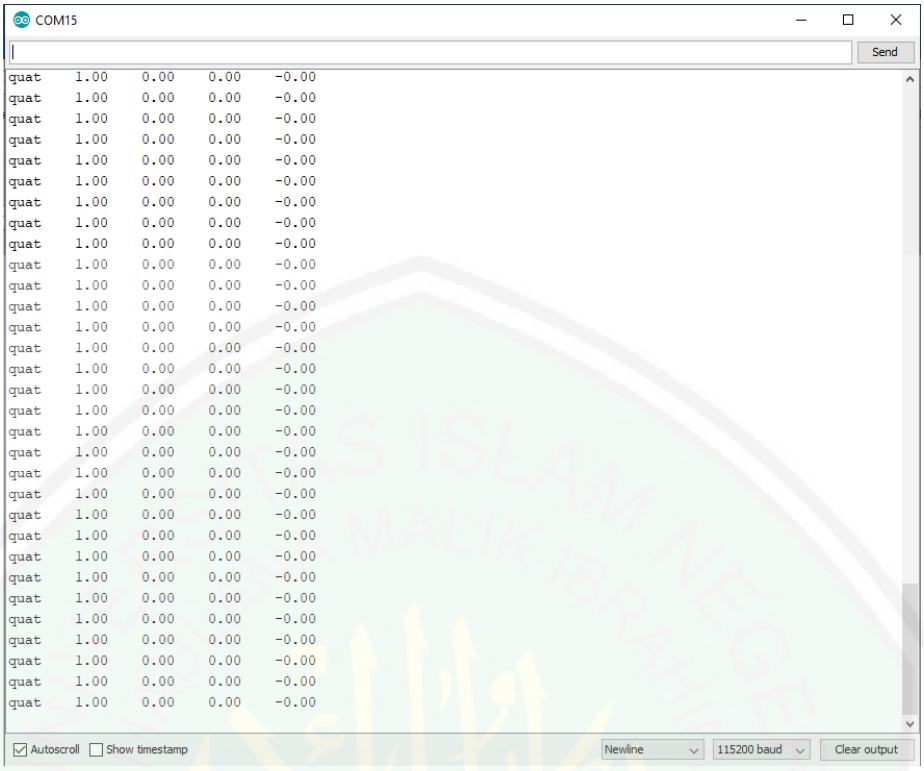
Gambar 4.2 merupakan hasil implementasi hardware robot sebagai objek test fungsi *remote control* sebelumnya yang terdiri dari : Microcontroller Arduino Uno sebagai pusat komputasi. *Driver Motor* yang berfungsi sebagai penerjemah perintah dari microcontroller untuk mengaktikfkan komponen penggerak Robot yakni MotorDC. 2 buah DC Motor sebagai alat penggerak robot.

4.2. Pengujian *Hardware*

Pengujian *hardware* ini dilakukan oleh 1 orang tester dengan cara pemakaian pada setiap alat untuk mengetahui apakah alat mampu bekerja dengan baik atau tidak.

4.2.1 Uji Coba Sensor *Accelerometer & Gyroscope* MPU6050

Pengujian berisi ujicoba hardware Sensor *Accelerometer & Gyroscope* MPU6050. Hal ini perlu dilakukan untuk mengetahui bahwa kondisi *haradware* dalam keadaan baik.

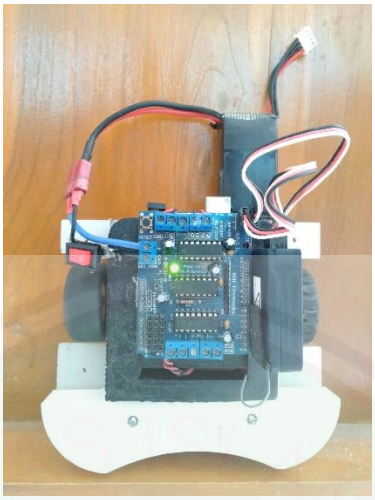


Gambar 4. 3 Hasil nilai pembacaan sensor MPU6050 terhadap gerakan tangan

Gambar 4.3 merupakan hasil test pembacaan sensor MPU6050 terhadap gerakan tangan. Tangan diposisikan sedemikian rupa untuk menunjukkan posisi diam, maju, mundur, belok kiri dan belok kanan. Dari gambar tersebut terlihat bahwa sensor mampu membaca setiap gerakan dengan menghasilkan nilai quaternion yang bervariasi sesuai dengan tiap-tiap gerakan.

4.2.2 Uji Coba Motor DC Pada Robot

Robot memiliki penggerak utama yakni motor DC. Komponen ini perlu diuji coba apakah dapat berfungsi dengan baik ketika dirakit ke dalam komponen *body* dari robot tersebut.



Gambar 3. 15 Uji coba robot

Uji coba dilakukan seperti Gambar 3.21. Robot diprogram sederhana untuk melakukan gerakan beruntun yakni maju, mundur, belok kiri dan belok kanan. Dari hasil uji coba, robot mampu menjalankan seluruh intruksi program tanpa kendala.

4.3. Pengujian Sistem

Pengujian Sistem ini dilakukan oleh 1 orang tester dengan cara penggunaan alat mulai dari tahap kalibrasi hingga klasfikasi *Neural Network Backpropagation*. Data uji dimasukkan sebanyak 10 kali pada tiap *gesture* tangan.

4.3.1. Kalibrasi

Seperti penjelasan pada sub bab 3.2.1, proses ini mengambil nilai tangan pada tiap posisi untuk digunakan sebagai data latih *NN Backpropagation* pada proses selanjutnya.

Gambar 4. 4 Proses pengambilan nilai posisi tangan diam

Gambar 4.4 merupakan proses pembacaan nilai posisi tangan diam. Bentuk data yang diambil pada proses ini sebagai berikut :

[id posisi, quaternion w, quaternion x, quaternion y, quaternion z] (4.1)

Id posisi merupakan nilai yang digunakan sebagai petunjuk posisi tangan. Nilai ini digunakan sebagai data latih output pada proses *Training NN Backpropagation*. Sedangkan nilai sisanya merupakan nilai quaternion yang merepresentasikan posisi tangan dalam bentuk 3 dimensi. Nilai ini digunakan sebagai data latih *input* pada proses *Training NN Backpropagation*. Berikut tabel hasil pengambilan nilai posisi tangan diam.

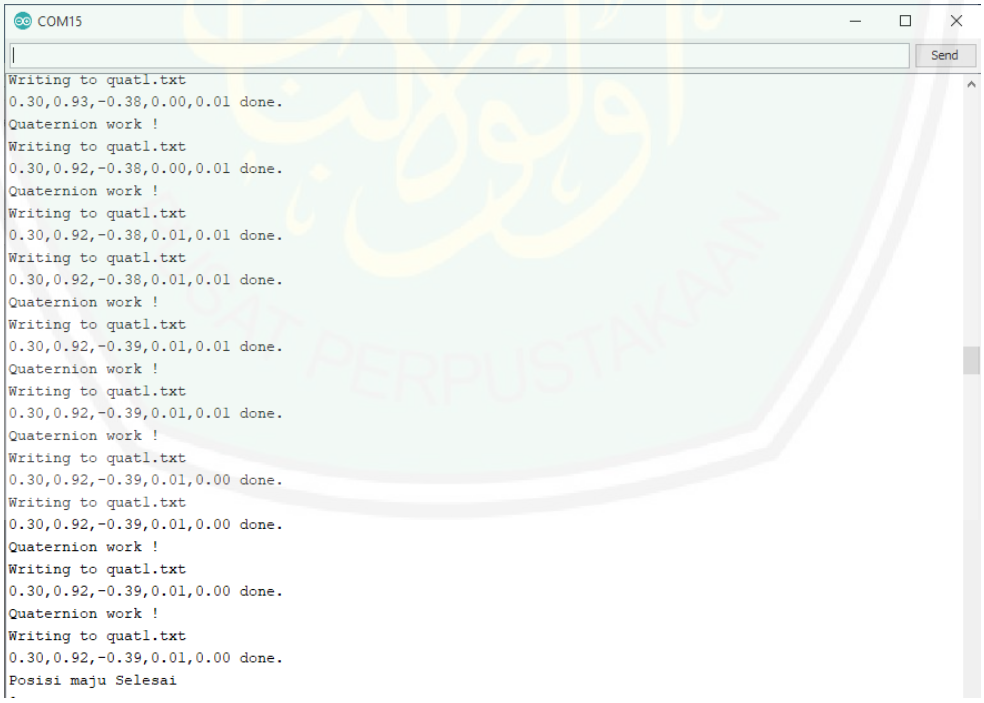
Tabel 4. 1 Nilai posisi tangan diam

W	X	Y	Z
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02

W	X	Y	Z
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02

W	X	Y	Z
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02

W	X	Y	Z
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02
1	0.01	0.01	-0.02



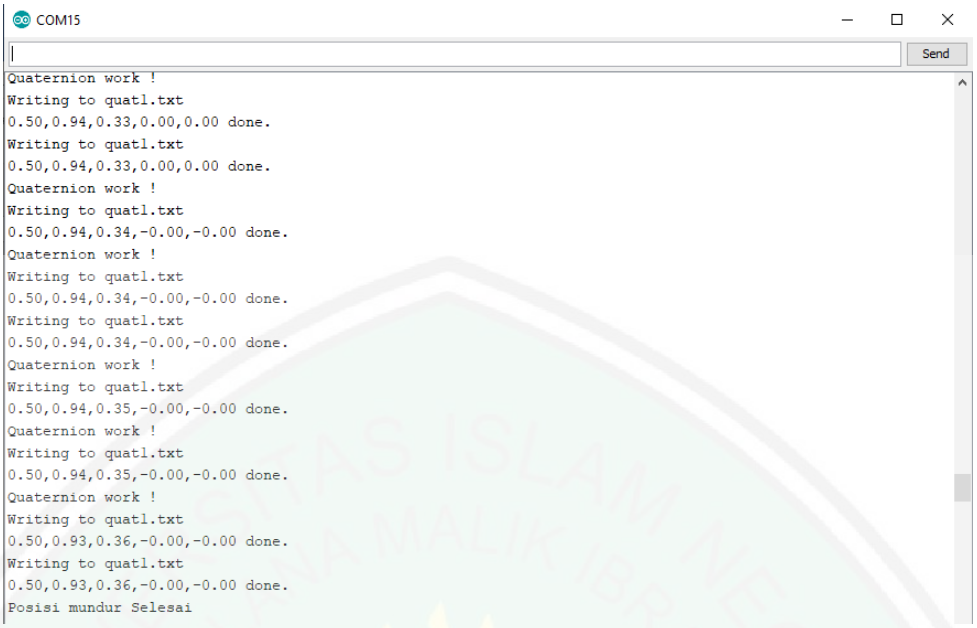
Gambar 4. 5 Proses pengambilan nilai posisi tangan maju

Gambar 4.5 merupakan proses pembacaan nilai posisi tangan maju. Berikut tabel hasil pengambilan nilai posisi tangan maju.

Tabel 4. 2 Nilai posisi tangan maju

W	X	Y	Z
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.35	0	0.01
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01

W	X	Y	Z
0.94	-0.35	0	0.01
0.94	-0.35	0	0.01
0.93	-0.35	0	0.01
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.36	0	0.01
0.93	-0.37	0	0.01
0.93	-0.37	0	0.01
0.93	-0.37	0	0.01
0.93	-0.38	0	0.01
0.92	-0.38	0	0.01
0.92	-0.38	0.01	0.01
0.92	-0.38	0.01	0.01
0.92	-0.39	0.01	0.01
0.92	-0.39	0.01	0.01
0.92	-0.39	0.01	0
0.92	-0.39	0.01	0
0.92	-0.39	0.01	0
0.92	-0.39	0.01	0



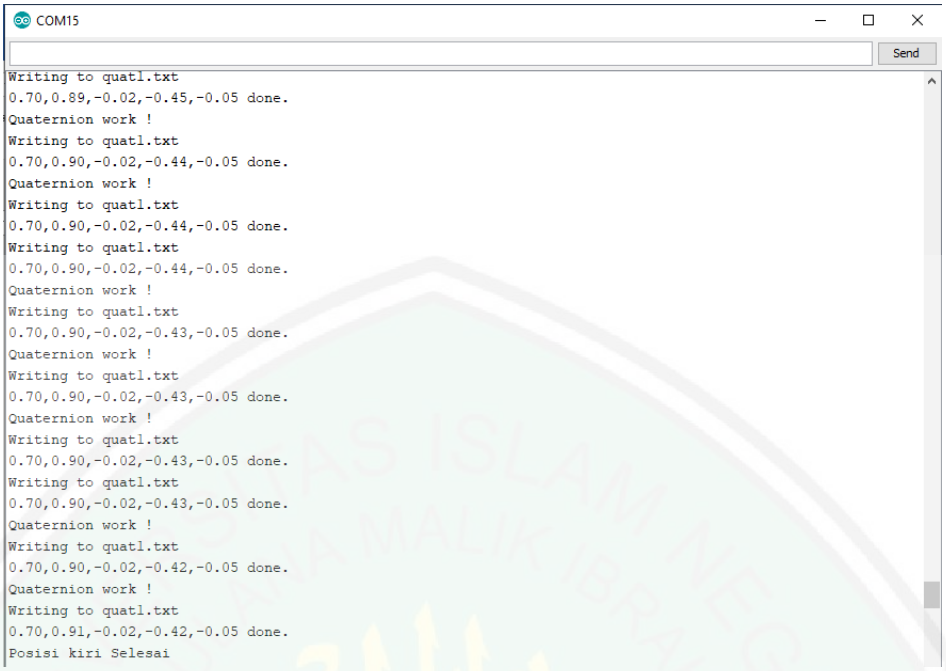
Gambar 4. 6 Proses pengambilan nilai posisi tangan mundur

Gambar 4.6 merupakan proses pembacaan nilai posisi tangan maju. Berikut tabel hasil pengambilan nilai posisi tangan mundur.

Tabel 4. 3 Nilai posisi tangan mundur

W	X	Y	Z
0.92	0.4	0	0.02
0.92	0.4	0	0.02
0.91	0.4	0.01	0.02
0.91	0.41	0.01	0.02
0.91	0.4	0.01	0.02
0.91	0.4	0.01	0.02
0.91	0.4	0.01	0.02
0.92	0.4	0.01	0.02
0.92	0.4	0.01	0.02
0.92	0.4	0.01	0.02
0.92	0.4	0.01	0.01
0.92	0.39	0.01	0.01
0.92	0.39	0	0.01
0.92	0.39	0	0.01
0.92	0.38	0	0.01
0.92	0.38	0	0.01
0.93	0.38	0	0.01
0.93	0.38	0	0.01
0.93	0.37	0	0.01
0.93	0.37	0	0.01

W	X	Y	Z
0.93	0.36	0	0.01
0.93	0.36	0	0.01
0.93	0.35	0	0.01
0.94	0.35	0	0.01
0.94	0.34	0	0.01
0.94	0.34	0	0.01
0.94	0.34	0	0.01
0.94	0.33	0	0
0.94	0.33	0	0
0.94	0.33	0	0
0.94	0.33	0	0
0.94	0.33	0	0
0.94	0.33	0	0
0.94	0.34	0	0
0.94	0.34	0	0
0.94	0.34	0	0
0.94	0.35	0	0
0.94	0.35	0	0
0.93	0.36	0	0
0.93	0.36	0	0



Gambar 4. 7 Proses pengambilan nilai posisi tangan belok kiri

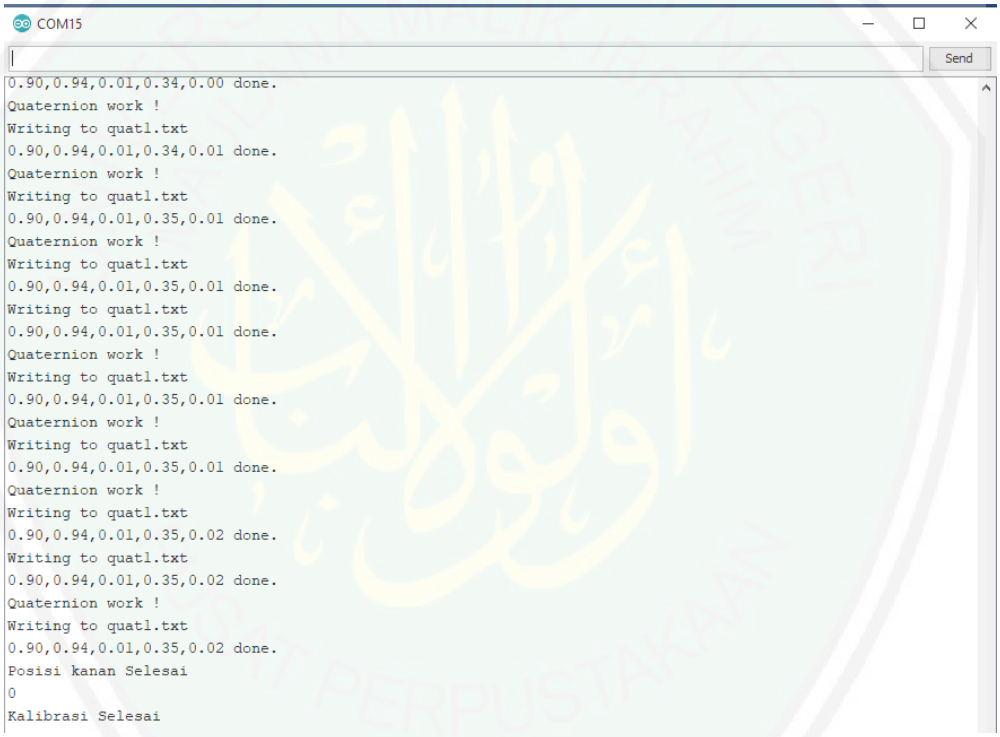
Gambar 4.7 merupakan proses pembacaan nilai posisi tangan belok kiri. Berikut tabel hasil pengambilan nilai posisi tangan belok kiri.

Tabel 4. 4 Tabel nilai posisi tangan belok kiri

W	X	Y	Z	W	X	Y	Z
0.9	-0.01	-0.43	-0.04	0.89	-0.01	-0.46	-0.04
0.9	-0.01	-0.43	-0.04	0.89	-0.01	-0.46	-0.04
0.9	-0.01	-0.43	-0.04	0.89	-0.01	-0.46	-0.04
0.9	-0.01	-0.44	-0.04	0.89	-0.01	-0.46	-0.04
0.9	-0.01	-0.44	-0.04	0.89	-0.01	-0.46	-0.04
0.9	-0.01	-0.44	-0.04	0.88	-0.01	-0.46	-0.04
0.89	-0.01	-0.44	-0.04	0.88	-0.01	-0.46	-0.04
0.89	-0.01	-0.45	-0.04	0.88	-0.01	-0.46	-0.04
0.89	-0.01	-0.45	-0.04	0.88	-0.01	-0.46	-0.04
0.89	-0.01	-0.45	-0.04	0.89	-0.01	-0.46	-0.04
0.89	-0.01	-0.46	-0.04	0.89	-0.01	-0.46	-0.04
0.89	-0.01	-0.46	-0.04	0.89	-0.01	-0.46	-0.04
0.89	-0.01	-0.46	-0.04	0.89	-0.01	-0.46	-0.04

W	X	Y	Z
0.89	-0.01	-0.46	-0.04
0.89	-0.01	-0.45	-0.05
0.89	-0.02	-0.45	-0.05
0.89	-0.02	-0.45	-0.05
0.89	-0.02	-0.45	-0.05
0.9	-0.02	-0.44	-0.05
0.9	-0.02	-0.44	-0.05

W	X	Y	Z
0.9	-0.02	-0.44	-0.05
0.9	-0.02	-0.43	-0.05
0.9	-0.02	-0.43	-0.05
0.9	-0.02	-0.43	-0.05
0.9	-0.02	-0.43	-0.05
0.9	-0.02	-0.42	-0.05
0.91	-0.02	-0.42	-0.05



Gambar 4. 8 Proses pengambilan nilai posisi tangan belok kanan

Gambar 4.8 merupakan proses pembacaan nilai posisi tangan belok kanan. Berikut tabel hasil pengambilan nilai posisi tangan belok kanan.

Tabel 4. 5 Tabel nilai posisi tangan belok kanan

W	X	Y	Z
0.94	0.02	0.34	0.02
0.94	0.02	0.34	0.02

W	X	Y	Z
0.94	0.02	0.34	0.02
0.94	0.02	0.34	0.02

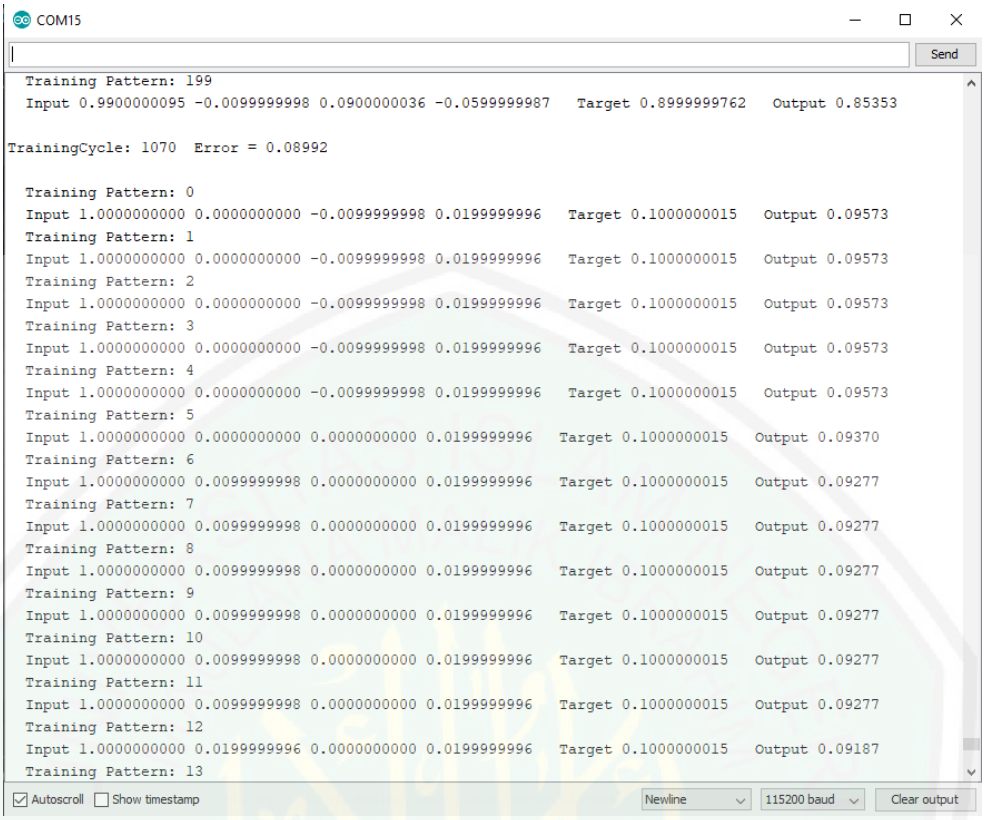
W	X	Y	Z
0.94	0.02	0.34	0.02
0.94	0.02	0.34	0.02
0.94	0.02	0.34	0.02
0.94	0.02	0.33	0.02
0.94	0.02	0.33	0.02
0.94	0.02	0.33	0.02
0.94	0.02	0.33	0.02
0.94	0.02	0.33	0.02
0.94	0.02	0.33	0.02
0.94	0.02	0.33	0.02
0.94	0.01	0.33	0.02
0.94	0.01	0.33	0.02
0.94	0.01	0.33	0.02
0.94	0.01	0.33	0.02
0.94	0.01	0.33	0.01
0.94	0.01	0.33	0.01
0.94	0.01	0.33	0.01
0.94	0.01	0.33	0.01

W	X	Y	Z
0.94	0.01	0.33	0.01
0.94	0.01	0.34	0.01
0.94	0.01	0.34	0.01
0.94	0.01	0.34	0.01
0.94	0.01	0.34	0.01
0.94	0.01	0.34	0
0.94	0.01	0.34	0
0.94	0.01	0.34	0
0.94	0.01	0.34	0
0.94	0.01	0.34	0.01
0.94	0.01	0.35	0.01
0.94	0.01	0.35	0.01
0.94	0.01	0.35	0.01
0.94	0.01	0.35	0.01
0.94	0.01	0.35	0.01
0.94	0.01	0.35	0.02
0.94	0.01	0.35	0.02
0.94	0.01	0.35	0.02

Seluruh nilai ini disimpan dalam bentuk text kedalam SDCard untuk digunakan kemudian pada proses *Training NN*.

4.3.2. *Training*NN

Proses *Training* dilakukan terlebih dahulu sebelum fungsi klasifikasi dapat dilakukan. Data seluruh posisi tangan yang tersimpan di *sd card* pada proses sebelumnya digunakan sebagai parameter input dan output *Training* pada arsitektur NN di *microcontroller*. Proses *Training* berjalan hingga nilai *error* mencapai batas yang ditentukan yakni sebesar 0,09. Proses *Training* seperti yang terlihat pada Gambar 4.9.

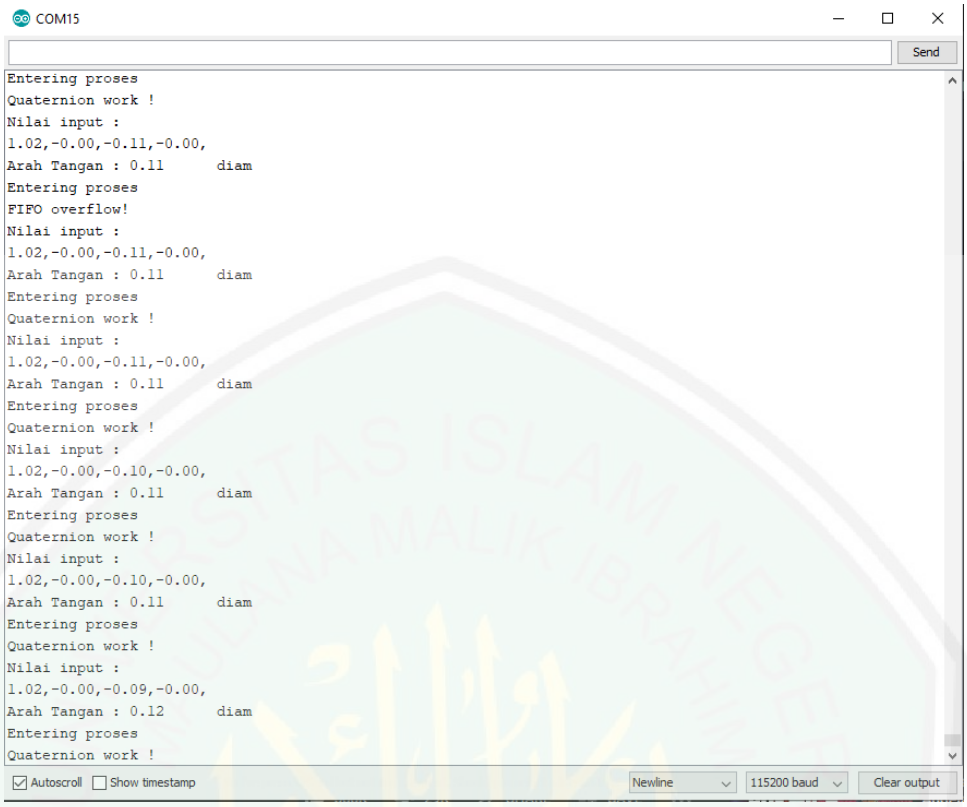


Gambar 4. 9 Proses *Training* pada *microcontroller*

Pada gambar 4.9 terlihat bahwa proses *Training* berhenti (sukses) pada epoch ke **1070** dengan nilai error sebesar **0,08992**.

4.3.3. Klasifikasi NN

Pada proses ini *User* memeragakan *gesture* arah kendali (diam, maju, undur, belok kiri, belok kanan) untuk mengendalikan robot seperti *gesture* yang dipakai pada proses kalibrasi sebelumnya. Data *gesture* ditangkap oleh sensor *gyroscope* MPU6050 dan diklasifikasi menggunakan metode *NN Backpropagation* yang telah dilatih pada proses *Training* sebelumnya. Proses klasifikasi menghasilkan output antara 0-1 dengan 5 kelas yakni diam, maju, mundur, kiri dan kanan. Proses klasifikasi dari *input user* secara *realtime* seperti yang tertera pada Gambar 4.10



Gambar 4. 10 Proses klasifikasi *input user* secara *realtime*

Tabel hasil output klasifikasi sebanyak 50 kali percobaan pada tiap-tiap *gesture* akan dijabarkan pada tabel dibawah. Presentasi akurasi dari masing-masing data percobaan hasil output kemudian dihitung menggunakan rumus berdasarkan Wachtmeister, Henke, dan Höök (2018).

$$akurasi = \frac{Jumlah\ State\ yang\ benar}{Jumlah\ state\ yang\ diuji} \times 100\% \qquad (4.2)$$

Tabel 4. 6 Tabel hasil output klasifikasi *gesture* diam

W	X	Y	Z	OUTPUT	HASIL KLASIFIKASI	HASIL SEHARUSNYA	KESESUAIAN
0.99	0	0.03	0	0.18	maju	diam	Tidak Sesuai
1	0	0.01	0	0.15	maju	diam	Tidak Sesuai
1.01	0	-0.04	0	0.12	diam	diam	Sesuai
1.02	0	-0.06	-0.01	0.12	diam	diam	Sesuai
1.02	0	-0.08	-0.01	0.11	diam	diam	Sesuai

W	X	Y	Z	OUTPUT	HASIL KLASIFIKASI	HASIL SEHARUSNYA	KESESUAIAN
1.02	0	-0.1	-0.01	0.11	diam	diam	Sesuai
1.02	0	-0.09	-0.01	0.11	diam	diam	Sesuai
1.03	0	-0.11	-0.01	0.1	diam	diam	Sesuai
1.03	0	-0.13	-0.01	0.1	diam	diam	Sesuai
1.03	0	-0.14	-0.01	0.1	diam	diam	Sesuai
1.03	0	-0.12	-0.01	0.1	diam	diam	Sesuai
1.03	0	-0.1	-0.01	0.11	diam	diam	Sesuai
1.04	0	-0.15	-0.01	0.1	diam	diam	Sesuai
1.04	0	-0.16	-0.01	0.1	diam	diam	Sesuai
1.04	0	-0.17	-0.01	0.1	diam	diam	Sesuai

Hasil percobaan pada Tabel 4.6, *user* diminta untuk menggerakan tangan untuk memberi perintah diam pada robot. Hasil pengujian mendapatkan akurasi seperti berikut.

$$akurasi = \frac{13}{15} \times 100\% = 86{,}7\ \% \qquad (4.3)$$

Tabel 4. 7 Tabel hasil output klasifikasi *gesture* maju

W	X	Y	Z	OUTPUT	HASIL KLASIFIKASI	HASIL SEHARUSNYA	KESESUAIAN
0.99	0.01	-0.14	0.06	0.21	maju	maju	sesuai
0.99	0.02	-0.14	0.06	0.21	maju	maju	sesuai
0.99	0.03	-0.14	0.06	0.21	maju	maju	sesuai
0.99	0.03	-0.13	0.06	0.21	maju	maju	sesuai
0.99	0.01	-0.13	0.06	0.2	diam	maju	tidak sesuai
0.99	0.03	-0.12	0.06	0.2	maju	maju	sesuai
1	0.02	0.04	0.04	0.2	maju	maju	sesuai
1	0.02	0.05	0.04	0.21	maju	maju	sesuai
1	0.02	0.05	0.03	0.21	maju	maju	sesuai
1	0.02	0.06	0.03	0.21	maju	maju	sesuai
0.99	0	0.06	0	0.21	diam	maju	tidak sesuai

W	X	Y	Z	OUTPUT	HASIL KLASIFIKASI	HASIL SEHARUSNYA	KESESUAIAN
1	0.02	0.07	0.03	0.22	maju	maju	sesuai
1	0.01	0.08	0.03	0.23	maju	maju	sesuai
0.98	0.01	0.09	0.01	0.27	diam	maju	tidak sesuai
0.97	0.01	0.13	0.01	0.33	diam	maju	tidak sesuai

Hasil percobaan pada Tabel 4.7, ketika *user* menggerakan tangan untuk memberi perintah maju pada robot. Hasil pengujian mendapatkan akurasi seperti berikut.

$$akurasi = \frac{11}{15} \times 100\% = 73{,}3\ \% \qquad (4.4)$$

Tabel 4. 8 Tabel hasil output klasifikasi *gesture* mundur

W	X	Y	Z	OUTPUT	HASIL KLASIFIKASI	HASIL SEHARUSNYA	KESESUAIAN
0.86	-0.52	0.02	0.01	0.43	mundur	mundur	sesuai
0.86	-0.51	0.02	0.01	0.41	mundur	mundur	sesuai
0.86	-0.51	0.01	0.01	0.41	diam	mundur	tidak sesuai
0.86	-0.5	0.01	0.01	0.4	diam	mundur	tidak sesuai
0.87	-0.5	0.01	0.01	0.4	diam	mundur	tidak sesuai
0.87	-0.5	0.02	0.01	0.41	mundur	mundur	sesuai
0.93	0.52	0.05	0.07	0.59	mundur	mundur	sesuai
0.93	0.51	0.05	0.07	0.57	mundur	mundur	sesuai
0.93	0.49	0.05	0.07	0.55	mundur	mundur	sesuai
0.93	0.48	0.05	0.07	0.54	mundur	mundur	sesuai
0.94	0.66	0.05	0.08	0.58	mundur	mundur	sesuai
0.94	0.67	0.05	0.08	0.6	mundur	mundur	sesuai
0.94	0.47	0.05	0.07	0.53	mundur	mundur	sesuai
0.94	0.45	0.04	0.07	0.51	mundur	mundur	sesuai
0.94	0.44	0.04	0.07	0.5	mundur	mundur	sesuai

Hasil percobaan pada Tabel 4.8, ketika *user* menggerakan tangan untuk memberi perintah mundur pada robot. Hasil pengujian mendapatkan akurasi seperti berikut.

$$akurasi = \frac{12}{15} \times 100\% = 80\ \% \quad (4.5)$$

Tabel 4. 9 Tabel hasil output klasifikasi *gesture* belok kiri

W	X	Y	Z	OUTPUT	HASIL KLASIFIKASI	HASIL SEHARUSNYA	KESESUAIAN
0.87	-0.05	-0.56	0.03	0.76	kiri	kiri	sesuai
0.87	-0.06	-0.59	0.03	0.79	kiri	kiri	sesuai
0.87	-0.06	-0.58	0.05	0.79	kiri	kiri	sesuai
0.88	-0.03	-0.53	0.03	0.73	kiri	kiri	sesuai
0.88	-0.06	-0.56	0.05	0.77	kiri	kiri	sesuai
0.89	-0.02	-0.49	0.03	0.68	kiri	kiri	sesuai
0.89	-0.05	-0.53	0.05	0.73	kiri	kiri	sesuai
0.9	0.01	-0.45	0.03	0.63	kiri	kiri	sesuai
0.9	-0.05	-0.5	0.06	0.69	kiri	kiri	sesuai
0.9	0.02	-0.43	0.07	0.65	kiri	kiri	sesuai
0.9	0.02	-0.43	0.07	0.66	kiri	kiri	sesuai
0.9	0.03	-0.44	0.07	0.67	kiri	kiri	sesuai
0.91	0.02	-0.41	0.07	0.62	kiri	kiri	sesuai
0.94	0.01	0.27	0.02	0.71	diam	kiri	tidak sesuai
0.95	0.01	0.24	0.02	0.62	diam	kiri	tidak sesuai

Hasil percobaan pada Tabel 4.9, ketika *user* menggerakan tangan untuk memberi perintah belok kiri pada robot. Hasil pengujian mendapatkan akurasi seperti berikut.

$$akurasi = \frac{13}{15} \times 100\% = 86{,}7\ \% \quad (4.6)$$

Tabel 4. 10 Tabel hasil output klasifikasi *gesture* belok kanan

W	X	Y	Z	OUTPUT	HASIL KLASIFIKASI	HASIL SEHARUSNYA	KESESUAIAN
0.82	-0.14	-0.78	0.03	0.9	kanan	kanan	sesuai
0.81	-0.14	-0.78	0.03	0.91	kanan	kanan	sesuai
0.81	-0.13	-0.78	0.03	0.91	kanan	kanan	sesuai
0.82	-0.14	-0.77	0.03	0.9	kanan	kanan	sesuai
0.81	-0.13	-0.77	0.03	0.91	kanan	kanan	sesuai
0.81	-0.12	-0.77	0.03	0.91	kanan	kanan	sesuai
0.83	-0.14	-0.76	0.03	0.9	kanan	kanan	sesuai
0.81	-0.12	-0.76	0.03	0.91	kanan	kanan	sesuai
0.81	-0.11	-0.76	0.03	0.9	kanan	kanan	sesuai
0.83	-0.14	-0.75	0.03	0.89	kanan	kanan	sesuai
0.81	-0.11	-0.75	0.03	0.9	kanan	kanan	sesuai
0.83	-0.13	-0.74	0.03	0.88	kanan	kanan	sesuai
0.81	-0.11	-0.74	0.03	0.9	kanan	kanan	sesuai
0.86	-0.08	-0.62	0.03	0.82	diam	kanan	tidak sesuai
0.9	0.01	0.42	0.03	0.9	diam	kanan	tidak sesuai

Hasil percobaan pada Tabel 4.10, ketika *user* menggerakan tangan untuk memberi perintah belok kanan pada robot. Hasil pengujian mendapatkan akurasi seperti berikut.

$$akurasi = \frac{13}{15} \times 100\% = 86{,}7\ \% \tag{4.7}$$

Nilai klasifikasi tersebut kemudian dikirim ke robot melalui *radio frequency* menggunakan modul nRF24L01. Nilai digunakan oleh robot untuk mengidentifikasi perintah gerak. Hasil akurasi klasifikasi tiap *gesture* kemudian dihitung menggunakan metode MSE seperti pada persamaan 4.2 berikut.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(f_i - y_i)^2 \tag{4.8}$$

$$MSE\ akurasi = \frac{(87-100)^2 + (73-100)^2 + (80-100)^2 + (87-100)^2 + (87-100)^2}{5} * 100\% \quad (4.9)$$

$$MSE\ akurasi = \frac{414}{5} * 100\%$$

$$MSE\ akurasi = 82{,}8\%$$

4.4. Integrasi Sains dan Islam

Penelitian ini bertujuan untuk menemukan instrumentasi kontrol yang lebih mudah, alami bagi manusia. Metode metode yang digunkana pada penelitian ini yakni *Hand Gesture Recognition* dan *Neural Network Backpropagation* merupakan proses matematis dengan pendekatan cara kerja tubuh manusia untuk menyelesaikan masalah yang ditemui. Hal ini mengingatkan kita bahwa sesungguhnya manusia adalah makhluk cipataan Allah yang paling sempurna. Banyak sumber ilmu yang bisa dikaji bahkan belum terkuak dari kesempurnaan penciptaan manusia itu sediri. Struktur penyusun hingga cara kerja semua organ pembentuk manusia yang luar bisa kerap dijadikan contoh untuk menyelesaikan permasalahan permasalahan dalam kehidupan sehari-hari. Seperti yang tertulis dalam Al Quran :

﴿لَقَدْ خَلَقْنَا ٱلْإِنسَٰنَ فِىٓ أَحْسَنِ تَقْوِيمٍ ۝٤﴾

Artinya : Sungguh Kami telah menciptakan manusia dalam bentuk dan sifat yang sebaik-baiknya. Dalam tafsir Jalalyn (“Surat At-Tin Ayat 4” 2019) ayat ini dijelaskan sebagai berikut : (Sesungguhnya Kami telah menciptakan manusia) artinya semua manusia (dalam bentuk yang sebaik-baiknya) artinya baik bentuk atau pun penampilannya amatlah baik.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kendali gerak robot telah berhasil diimplementasikan dengan menggunakan *Hand Gesture Recognition* berbasis *Neural Network Backpropagation*. Robot mampu dikenadalikan sesuai dengan instruksi yang diberikan melalui isyarat tangan. Terdapat beberapa catatan mengenai penelitian ini yakni keberhasilan pengenalan insruksi *gesture* tangan sangat bergantung terhadap cara pengguna memberikan contoh instruksi *gesture* ketika pada proses latih. Robot mampu merespon hasil instruksi dengan cukup cepat. Gestur yang mampu dikenali oleh alat ini hanya 5 macam. Serta, nilai seluruh akurasi hasil klasifikasi tiap *gesture* mencapai 82,8%.

5.2 Saran

Robot sejenis ini masih mampu untuk melakukan pergerakan dengan kondisi yang lebih banyak. Selain arah maju, mundur, belok kiri dan belok kanan, robot bisa dikembangkan untuk mengenali parameter lain seperti kecepatan. Sehingga laju pergerakan robot tidak statis.

Kedepannya, untuk menambah keamanan, sistem kontrol ini bisa dirancang untuk mengenali gerakan orang tertentu. Sehingga alat ini mampu diatur untuk bisa digunakan hanya oleh pengguna yang sudah memasukkan preferensi gerakannya sendiri atau pengguna yang gerakannya telah terdaftar di alat ini.

Sistem komputasi kontrol robot ini menggunakan *microcontroller* STM32F10C dengan kapasitas memori yang terbatas, sehingga data latih yang mampu ditampung oleh *microcontroller* tersebut hingga ambang batas terakhir hanya 200 buah. Penggunaan jenis *microcontroller* lain dengan kapasitas yang lebih besar memungkinkan pengguna untuk menambah jumlah data latih yang mempengaruhi tingkat keberhasilan pengenalan *gesture* itu sendiri. Selain itu, penggunaan metode dan kecerdasan buatan jenis lain cukup menarik untuk diterapkan dalam penelitian ini.

DAFTAR PUSTAKA

"Arduino - Introduction." 2019. 2019. https://www.arduino.cc/en/Guide/Introduction.

"Arduino Playground - MPU-6050." 2019. 2019. https://playground.arduino.cc/Main/MPU-6050.

Diebel, James. 2006. "Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors," Oktober, 35.

Dipietro, L., A.M. Sabatini, dan P. Dario. 2008. "A Survey of Glove-Based Systems and Their Applications." *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38 (4): 461–82. https://doi.org/10.1109/TSMCC.2008.923862.

Falcao, Christianne, Ana Catarina Lemos, dan Marcelo Soares. 2015. "Evaluation of Natural User Interface: A Usability Study Based on the Leap Motion Device." *Procedia Manufacturing* 3: 5490–95. https://doi.org/10.1016/j.promfg.2015.07.697.

Gallacher, B.J., J.S. Burdess, dan A.J. Harris. 2001. "Principles of a Three-Axis Vibrating *Gyroscope* scope." *IEEE Transactions on Aerospace and Electronic Systems* 37 (4): 1333–43. https://doi.org/10.1109/7.976969.

Gandhi, Vaibhavi S, Akshay A Khond, Sanket N Raut, Vaishali A Thakur, dan Shabnam S Shaikh. 2014. "A Review of Various Gesture *Recognition* Techniques" 3 (9): 5.

Maharani, Devira Anggi, Hanif Fakhrurroja, Riyanto, dan Carmadi Machbub. 2018. "*Hand* Gesture *Recognition* Using K-Means Clustering and Support Vector Machine." Dalam *2018 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, 1–6. Penang: IEEE. https://doi.org/10.1109/ISCAIE.2018.8405435.

Oyedotun, Oyebade K., dan Adnan Khashman. 2017. "*Deep learning* in Vision-Based Static *Hand* Gesture *Recognition*." *Neural Computing and Applications* 28 (12): 3941–51. https://doi.org/10.1007/s00521-016-2294-8.

Premaratne, Prashan. 2014. *Human Computer Interaction Using Hand Gestures*. Cognitive Science and Technology. Singapore: Springer.

"Surat At-Tin Ayat 4." 2019. Tafsir AlQuran Online. 2019. https://tafsirq.com/permalink/ayat/6102.

"Understanding Quaternions | CH Robotics." 2019. 2019. http://www.chrobotics.com/library/understanding-quaternions.

Wachtmeister, Henrik, Petter Henke, dan Mikael Höök. 2018. "Oil projections in retrospect: Revisions, accuracy and current uncertainty." *Applied Energy*. https://doi.org/10.1016/j.apenergy.2018.03.013.

Xu, Ruize, Shengli Zhou, dan Wen J. Li. 2012. "MEMS *Accelerometer* Based Nonspecific-User *Hand* Gesture *Recognition*." *IEEE Sensors Journal* 12 (5): 1166–73. https://doi.org/10.1109/JSEN.2011.2166953.