TESIS- KI 142502

PENGENALAN BAHASA ISYARAT SIBI MENGGUNAKAN FITUR STATIS DAN DINAMIS *LMC* BERBASIS RB-L-GCNN

SUPRIA
NRP. 5114201059

DOSEN PEMBIMBING
Dr. Eng. Darlis Heru Murti, S.Kom., M.Kom.
Wijayanti Nurul Khotimah, S.Kom., M.Sc.

PROGRAM MAGISTER
BIDANG KEAHLIAN INTERAKSI, GRAFIK DAN SENI
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

THESES- KI 142502

SIBI SIGN LANGUAGE RECOGNITION USING STATIC AND DYNAMIC FEATURES OF LMC BASED ON RB-L-GCNN

SUPRIA
NRP. 5114201059

SUPERVISOR
Dr. Eng. Darlis Heru Murti, S.Kom., M.Kom.
Wijayanti Nurul Khotimah, S.Kom., M.Sc.

MASTER PROGRAM
INTERACTION, GRAPHICS AND ART
DEPARTMENT OF INFORMATICS ENGINEERING
FACULTY OF INFORMATION TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

Telah disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Komputer (M.Kom)
di
Institut Teknologi Sepuluh Nopember Surabaya

Oleh:
SUPRIA
NRP. 5114201059

Dengan Judul :
Pengenalan Bahasa Isyarat SIBI Menggunakan Fitur Statis dan Dinamis LMC Berbasis RB-LGCNN

Tanggal Ujian: 30 Juni 2016
Periode Wisuda: September 2016

Disetujui oleh:

Dr. Darlis Heru Murti, S.Kom, M.Kom
NIP. 19771217 200312 1 001 ..... (Pembimbing 1)

Wijayanti Nurul Khotimah, S.Kom., M.Sc
NIP. 19860312 201212 2 004 ..... (Pembimbing 2)

Dr. Eng. Nanik Suciati, S.Kom, M.Kom
NIP. 19710428 199412 2 001 ..... (Penguji 1)

Imam Kuswardayan, S.Kom, MT
NIP. 19761215 200312 1 001 ..... (Penguji 2)

Ridho Rahman Hariadi, S.Kom, M.Sc
NIP. 19870213 201404 1 001 ..... (Penguji 3)

Direktur Program Pasca Sarjana,

Prof.Ir. Djauhar Manfaat, M.Sc., Ph.D.
NIP. 19601202 198701 1 001

Pengenalan Bahasa Isyarat SIBI Menggunakan Fitur Statis dan Dinamis LMC Berbasis RB-L-GCNN

Nama	: Supria
NRP	: 5114201059
Dosen Pembimbing I	: Dr. Eng. Darlis Heru Murti, S.Kom, M.Kom
Dosen Pembimbing II	: Wijayanti Nurul Khotimah, S.Kom, M.Sc

ABSTRAK

Proses komunikasi antara penyandang tunarungu dan tunawicara dapat dipahami antar sesama dengan baik karena mereka sudah terbiasa sehari-harinya menggunakan bahasa isyarat. Namun sebagian besar orang normal akan kesulitan untuk memahami bahasa isyarat yang disampaikan oleh penyandang tunarungu dan tunawicara, begitu juga sebaliknya, penyandang tunarungu dan tunawicara akan kesulitan memahami bahasa yang disampaikan oleh orang normal. Untuk mengatasi masalah tersebut maka dibangun sebuah sistem pengenalan bahasa isyarat dengan menggunakan *Leap Motion Controller* (LMC). Pada penelitian sebelumnya, pengenalan bahasa isyarat *American Sign Language* (ASL) menggunakan LMC dengan menggunakan fitur yang bersifat statis berdasarkan pada KNN dan SVM memiliki akurasi pengenalan yang cukup baik.Namun metode tersebut hanya dapat mengenal bahasa isyarat yang bersifat statis. Padahal bahasa isyarat ada dua macam yaitu bahasa isyarat yang bersifat statis dan bahasa isyarat yang bersifat dinamis. Selain itu *Logarithmic Learning for Generalized Classifier Neural Network* (L-GCNN) merupakan metode yang handal dalam menangani klasifikasi data. Namun ketika L-GCNN digunakan pada data yang memiliki kelas yang banyak maka akan terjadi *overfitting* atau kesulitan dalam menentukan kelas pada data.

Pada penelitian ini diusulkan pengenalan bahasa isyarat SIBI yang mengkombinasikan fitur statis dan fitur dinamis dari LMC berdasarkan *Rule Based*L-GCNN (RB-L-GCNN). Dimana fitur statis dimanfaatkan untuk pengenalan bahasa isyarat yang bersifat statis, sedangkan fitur dinamis dimanfaatkan untuk mengenal bahasa isyarat yang bersifat dinamis. *Rule based* dimanfaatkan untuk mengurangi terjadinya *overfitting* pada metode klasifikasi L-GCNN.

Dari hasil pengujian yang dilakukan pengenalan bahasa isyarat SIBI dengan menggunakan kombinasi fitur statis dengan fitur dinamis dapat mengenal bahasa isyarat yang bersifat statis maupun bahasa isyarat yang bersifat dinamis. Sedangkan pembentukan *rule based* pada L-GCNN dapat meningkatkan akurasi pengenalan hingga 6.67%.

Kata kunci: Pengenalan bahasa isyarat, *leap motion controller*, fitur statis dan dinamis, *rule based*, L-GCNN.

[*Halaman ini sengaja dikosongkan*]

SIBI Sign Language Recognition Using Static and Dynamic Features of LMC Based on RB-L-GCNN

Name : Supria
Student Identity Number : 5114201059
Supervisor : Dr. Eng. Darlis Heru Murti, S.Kom, M.Kom
Co-Supervisor : Wijayanti Nurul Khotimah, S.Kom, M.Sc

ABSTRACT

The process of communication between the deaf and dumb people can be understood by each other well because they are already familiar to sign language. However, most of normal people will find it hard to understand sign language conveyed by the deaf and dumb people, and vice versa, the deaf and dumb people will have trouble to understand the language conveyed by normal people. To overcome these problems, we will develop a sign language recognition system using Leap Motion Controller (LMC). In previous research, the sign language recognition of American Sign Language (ASL) uses LMC that it uses the static features based on KNN and SVM that has recognition accuracy well. But, these methods can only recognize the static sign language. Where the sign language has two types, static sign language and dynamic sign language. Moreover Logarithmic Learning for Generalized Classifier Neural Network (L-GCNN) is a reliable methods to overcome data classification. But, when L-GCNN is used to data that have many classes, it will occur overfitting in determining the class of the data.

In this study, we propose the SIBI sign language recognition which combines static and dynamic features of the LMC based on Rule Based L-GCNN (RB-L-GCNN). The static features is used for the recognition of static sign language, and the dynamic features is used to recognize the dynamic sign language. Rule based is used to reduce the occurrence of overfitting in L-GCNN classification methods.

From the results of tests performed SIBIsign language recognition using a combination of static features with dynamic features can recognize static sign language or dynamic sign language. While the establishment of the rule based on L-GCNN can improve recognition accuracy up to 6.67%.

Keyword: Leap motion controller, L-GCNN, static and dynamic features, rule based, sign language recognition

[*Halaman ini sengaja dikosongkan*]

DAFTAR ISI

ABSTRAK vii
ABSTRACT ix
KATA PENGANTAR xi
DAFTAR ISI xiii
DAFTAR GAMBAR xv
DAFTAR TABEL xvii
BAB 1 PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 3
1.3 Batasan Masalah 3
1.4 Tujuan Penelitian 4
1.5 Manfaat penelitian 4
1.6 Kontribusi 4
1.7 Sistematika Penulisan 4
BAB 2 DASAR TEORI DAN KAJIAN PUSTAKA 7
2.1 Bahasa Isyarat 7
2.2 Leap Motion Controller (LMC) 8
2.2.1 Sistem Koordinat 8
2.2.2 *Motion Tracking Data* 9
2.2.2.1 *Hands* 9
2.2.2.2 *Arms* 10
2.2.2.3 *Fingers* 10
2.2.2.4 *Tools* 12
2.2.2.5 *Gestures* 12
2.2.2.6 *Motions* 12
2.3 Fitur Statis 13
2.4 Fitur Dinamis 16
2.5 *Decission Tree* 17
2.6 Logarithmic Learning for Generalized Classifier Neural Network (L-GCNN) 17
BAB 3 METODOLOGI PENELITIAN 25
3.1 Proses Awal 26
3.2 Ekstraksi Fitur 27
3.2.1 Fitur Statis 27
3.2.1.1 *Average Spread* 27
3.2.1.2 *Average Tri-Spread* 28
3.2.1.3 *Extended Distance* 30
3.2.2 Fitur Dinamis 31
3.3 Penggabungan Fitur Statis dan Dinamis 32
3.4 Normalisasi 32
3.5 Kalibrasi 33
3.6 *Rule Based* 33
3.7 Klasifikasi 35

3.7.1 Proses Pelatihan 35
3.7.2 Proses Uji Coba 36
BAB 4 HASIL UJI COBA DAN PEMBAHASAN 39
4.1 Lingkungan Uji Coba 39
4.2 Data Uji Coba 39
4.3 Skenario Uji Coba dan Evaluasi 40
4.3.1 Hasil Skenario Uji Coba A dan Analisa 41
4.3.1.1 Hasil Skenario Uji Coba A1 dan Analisa 41
4.3.1.2 Hasil Skenario Uji Coba A2 dan Analisa 46
4.3.2 Hasil Skenario Uji Coba B dan Analisa 50
4.3.2.1 Hasil Skenario Uji Coba B1 dan Analisa 51
4.3.2.2 Hasil Skenario Uji Coba B2 dan Analisa 53
4.4 Pembahasan Hasil 57
BAB 5 KESIMPULAN DAN SARAN 59
5.1 Kesimpulan 59
5.2 Saran 59
DAFTAR PUSTAKA 61
LAMPIRAN 1 63
LAMPIRAN 2 71
BIODATA PENULIS 73

DAFTAR GAMBAR

Gambar 2.1 Pandangan LMC terhadap tangan (Leap Motion 2015). 8
Gambar 2.2 Sistem koordinat *leap motion*..... 9
Gambar 2.3 *Palm* normal dan arah vector. 10
Gambar 2.4 Posisi ujung jari dan arah vektor. 11
Gambar 2.5 Tulang tulang pada jari tangan. 11
Gambar 2.6 *Tools*. 12
Gambar 2.7 *Motions*. 13
Gambar 2.8 Capture tangan menggunakan LMC. 15
Gambar 2.9 Proses *quantization* (A) Contoh pergerakan dalam bidang *XOZ*, (B) Acuan *quantization*, (C) Hasil *quantization*. 17
Gambar 2.10 Arsitektur L-GCNN. 18
Gambar 3.1 Sistem pengenalan bahasa isyarat SIBI. 25
Gambar 3.2 Tahap proses awal. 26
Gambar 3.3 Perhitungan fitur *average spread* 28
Gambar 3.4 Perhitungan fitur *average tri-spread.* 29
Gambar 3.5 Perhitungan fitur *extended distance* 30
Gambar 3.6 Perhitungan fitur dinamis 31
Gambar 3.7 Pembagian *rule* berdasarkan *Decission tree.* 34
Gambar 3.8 Proses pelatihan dataset. 36
Gambar 3.9 Proses uji coba. 37
Gambar 4.1 Hasil uji coba fitur statis dan RB-L-GCNN 42
Gambar 4.2 Hasil uji coba fitur statis dan L-GCNN 43
Gambar 4.3 Hasil uji coba fitur statis dan *Decission tree* 44
Gambar 4.4 Hasil uji coba fitur statis dan SVM dengan kernel linier. 44
Gambar 4.5 Hasil uji coba fitur statis dan KNN dengan K=1. 45
Gambar 4.6Hasil perbandingan skenario A1 45
Gambar 4.7 Hasil uji coba kombinasi fitur statis dengan dinamis dan RB-L-GCNN 46
Gambar 4.8 Hasil uji coba kombinasi fitur statis dengan dinamis dan L-GCNN. 47
Gambar 4.9 Hasil uji coba kombinasi fitur statis dengan dinamis dan *Decission tree*. 48
Gambar 4.10 Hasil uji coba kombinasi fitur statis dengan dinamis dan SVM dengan kernel linier. 48
Gambar 4.11 Hasil uji coba kombinasi fitur statis dengan dinamis dan KNN dengan K=1. 49
Gambar 4.12Hasil perbandingan skenario A2 49
Gambar 4.13Hasil perbandingan skenario A1 dan A2 50
Gambar 4.14 Hasil skenario uji coba B1 tahap pertama 51
Gambar 4.15 Hasil skenario uji coba B1 tahap kedua 52
Gambar 4.16 Hasil perbandingan akurasi metode L-GCNN dan RB-L-GCNN dengan menggunakan fitur statis. 53
Gambar 4.17 Hasil skenario uji coba B2 tahap pertama. 54
Gambar 4.18 Hasil skenario uji coba B2 tahap kedua 55

Gambar 4.19 Hasil perbandingan akurasi metode L-GCNN dan RB-L-GCNN dengan menggunakan kombinasi fitur statis dan fitur dinamis. 56
Gambar 4.20 Hasil skenario B1 dan B2 56

DAFTAR TABEL

Tabel 2.1 Satuan ukuran LMC 9
Tabel 2.2 Tipe *motion* 13
Tabel 3.1 Contoh kombinasi fitur statis dan dinamis..... 32
Tabel 3.2*Pseudocode* pembagian rule..... 34

[*Halaman ini sengaja dikosongkan*]

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Komunikasi antara manusia adalah suatu hal yang sangat penting bagi aktivitas kehidupan sehari-hari. Ada beberapa jenis komunikasi yaitu komunikasi secara lisan, tulisan, dan isyarat. Komunikasi secara isyarat biasanya digunakan oleh penyandang tunarungu dan cacat tunawicara. Sistem Isyarat Bahasa Indonesia (SIBI) adalah salah satu komunikasi bahasa isyarat yang dimiliki oleh negara Indonesia. SIBI dibangun dengan mengadopsi dari bahasa isyarat American Sign Language (ASL) yang dimiliki oleh negara Amerika.

Proses komunikasi antara penyandang tunarungu dan tunawicara dapat dipahami antar sesama dengan baik karena mereka sudah terbiasa sehari-harinya menggunakan bahasa isyarat. Namun untuk orang normal akan kesulitan untuk memahami bahasa isyarat yang disampaikan oleh penyandang tunarungu dan tunawicara karena ada perbedaan metode komunikasi, begitu juga sebaliknya, penyandang tunarungu dan tunawicara akan kesulitan memahami bahasa yang disampaikan oleh orang normal. Untuk itu dibutuhkan sebuah sistem yang dapat menerjemahkan perbedaan metode komunikasi antara komunikasi bahasa isyarat dengan komunikasi bahasa normal. Untuk menangani masalah tersebut maka dibangun sebuah sistem pengenalan bahasa isyarat.

Penelitian tentang pengenalan bahasa isyarat dibagi kedalam tiga kategori besar yaitu : berbasis computer visi(Zaki & Shaheen 2011), (Sharma & Verma 2015), (Zhou et al. 2016), berbasis sensor *glove*(Shukor et al. 2015), (Oz & Leu 2007) dan sensor *motion*, dan kombinasi antara kedua metode tersebut (Oz & Leu 2007). Pengenalan bahasa isyarat berbasiskomputer visi akan membutuhkan komputasi yang sangat komplek, karena memperhitungkan setiap piksel pada citra. *Human Computer Interaction* (HCI) telah memperkenalkan sistem yang efektif pada pengenalan *hand gesture* (Chen et al. 2015).

Sistem pengenalan bahasa isyarat ASL dengan menggunakan LMC berdasarkan pada sebuah metode *machine learning* yaitu *K-Nearest Neighbor*

(KNN) yang dibandingkan dengan *support vector machine* (SVM) pernah dilakukan (Chuan et al. 2014). Sistem tersebut menghasilkan akurasi pengenalan cukup baik untuk pengenalan bahasa isyarat yang bersifat statis. Beberapa fitur statis dariLMC telah digunakan yaitu: fitur *pinch strength, grab strength, average distance, average spread, average tri-spread, extended distance, dip-tip projection, orderX* dan *angle*. Namun sistem tersebut hanya dapat mengenal bahasa isyarat yang bersifat statis dan tidak dapat mengenal bahasa isyarat yang bersifat dinamis seperti huruf *J* dan *Z* dengan baik. Selain itu *Rapid recognition* (pengenalan cepat) dari *dynamic hand gesture* dengan menggunakan LMC berdasarkan SVM diusulkan untuk pengenalan gerakan yang berbentuk huruf atau yang berbentuk *gesture*(Chen et al. 2015). Sistem tersebut memanfaatkan fitur *gesture* pada *leap motion* untuk pengenalan cepat *dynamic handgesture*. Namun sistem tersebut hanya dapat mengenal gerakan tangan.

Generalized classifier neural network (GCNN) merupakan metode klasifikasi yang diusulkan untuk melakukan klasifikasi dengan akurasi yang tinggi (Melis & Avci 2013). Pada proses klasifikasi beberapa dataset yang diuji, GCNN memiliki akurasi yang lebih baik dibandingkan dengan metode GRNN dan PNN. *Logarithmic learning for generalized classifier neural network* (L-GCNN) merupakan metode pengembangan dari GCNN yang bertujuan untuk mengurangi waktu yang dibutuhkan pada proses klasifikasi dan meningkatkan akurasi klasifikasi (Melis & Avci 2014). L-GCNN sangat handal dalam menangani klasifikasi data, bahkan memiliki akurasi yang lebih baik dan komputasinya lebih cepat dibandingkan dengan GCNN setelah dilakukan perbandingan pada proses klasifikasi terhadap beberapa dataset. Namun ketika L-GCNN digunakan pada data yang memiliki kelas yang banyak akan terjadi *overfitting* atau sulit untuk menentukan kelas pada data. Disisi lain, penambahan *rule based* pada proses klasifikasi berbasis *neural network* dapat mengurangi terjadinya *overfitting*(Khotimah et al. 2015).

Pada penelitian ini diusulkan sistem pengenalan bahasa isyarat SIBI yang mengkombinasikan fitur statisdan fitur dinamis yang didapatkan dari LMC berbasis *rule based*L-GCNN. Fitur statis *pinch strength, grab strength, average distance, average spread, average tri-spread, extended distance, dip-tip*

projection, orderX dan *angle* dimanfaatkan untuk pengenalan bahasa isyarat yang bersifat statis, sedangkan fitur *hand dynamic gesture* dimanfaatkan untuk pengenalan bahasa isyarat yang bersifat dinamis. *Rule based* dimanfaatkan untuk pengelompokan kelas dengan tujuan untuk mengurangi terjadinya *overfitting* dan meningkatkan akurasi klasifikasi pada L-GCNN. L-GCNN digunakan untuk melakukan klasifikasi antara data uji coba terhadap data latih atau data model dari fitur statis danfitur dinamis pada LMC. Metode yang diusulkan diharapkan dapat meningkatkan akurasi pengenalan bahasa isyarat yang bersifat statis maupun bahasa isyarat yang bersifat dinamis. Pengenalan bahasa isyarat yang efektif dan efisien dapat membantu memudahkan komunikasi antara orang penyandangtunarungu dan tunawicara.

1.2 Rumusan Masalah

Dari penjelasan latar belakang penelitian ini dapat disimpulkan beberapa rumusan masalah yaitu sebagai berikut:

1. Bagaimana mendapatkan fitur-fitur dari LMC untuk pengenalan bahasa isyarat yang bersifat statis maupun dinamis?
2. Bagaimana cara meningkatkan akurasi proses klasifikasi pada L-GCNN dengan menggunakan *rule based*?
3. Bagaimana cara meningkatkan kualitas hasil pengenalan bahasa isyarat?

1.3 Batasan Masalah

Adapun batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Bahasa isyarat yang digunakan adalah bahasa isyarat SIBI dengan huruf abjad (A-Z).
2. Hanya penerjemahan dari bahasa isyarat ke bahasa normal.
3. Dataset yang digunakan adalah dataset yang direkam sendiri menggunakan tangan penulis dengan jumlah 10 sampel untuk setiap huruf.

1.4 Tujuan Penelitian

Tujuan penelitian ini adalah mengusulkan sebuah metode pengenalan bahasa isyarat yang bersifat statis dan bahasa isyarat yang bersifat dinamis dengan menggunakan LMC berdasarkan pada *rule based*L-GCNN dengan memanfaatkan fitur statis dan fitur dinamis yang didapatkan dari LMC.

1.5 Manfaat penelitian

Metode yang diusulkan diharapkan dapat mengenal bahasa isyarat yang bersifat statis maupun dinamis dan menghasilkan akurasi yang tinggi pada pengenalan bahasa isyarat SIBI. Proses pengenalan bahasa isyarat yang tepat dapat membantu proses komunikasi antara penyandang tunarungu dan tunawicara.

1.6 Kontribusi

Adapun kontribusi pada penelitian ini adalah mengusulkan sebuah metode pengenalan bahasa isyarat SIBI yang mengkombinasikanfitur statis dan fitur dinamis yang didapatkan dari LMC berbasis *rule based* pada L-GCNN. Pemanfaatan beberapa fitur statis *average spread, average tri-spread, extended distance,* dan fitur *hand dynamicgesture* pada LMC dapat digunakan untuk pengenalan bahasa isyarat yang bersifat statis maupun dinamis, serta pembentukan *rule based* pada L-GCNN dapat mengurangi terjadinya *overfitting* dan meningkatkan akurasi pengenalan.

1.7 Sistematika Penulisan

Pada laporan penelitian ini akan dibagi kedalam beberapa bab, adapun isi dari bab-bab tersebut adalah:

1. BAB I Pendahuluan, pada bab ini dijelaskan tentang latar belakang, permasalahan, rumusan masalah, batasan masalah, tujuan, manfaat, kontribusi dan sistematika penulisan.
2. BAB II Dasar Teori dan Kajian Pustaka, pada bab ini akan dibahas beberapa dasar teori yang mendukung penelitian tersebut.
3. BAB III Metode Penelitian, pada bab ini akan dibahas tentang metodologi penelitian yang dilakukan.

4. BAB IV Ujicoba dan Analisa Hasil, pada bab akan dibahas tentang ujicoba yang dilakukan terhadap metode penelitian yang diusulkan, serta hasil ujicoba penelitian.
5. BAB V Kesimpulan dan Saran, pada bab ini akan dibahas tentang hasil kesimpulan dari metode penelitian yang diusulkan dan saran untuk pengembangannya.

[*Halaman ini sengaja dikosongkan*]

BAB 2
DASAR TEORI DAN KAJIAN PUSTAKA

2.1 Bahasa Isyarat

Bahasa isyarat adalah salah satu bahasa komunikasi yang dilakukan dengan menggunakan pergerakan tangan, pergerakan tubuh, atau ekpresi wajah. Bahasa isyarat ini merupakan bahasa yang digunakan oleh penyandang tunarungu dan tunawicara untuk melakukan komunikasi antar sesama. Belum ada bahasa isyarat internasional karena bahasa isyarat setiap negara memiliki perbedaan. Beberapa bahasa isyarat nasional yang ada sampai saat ini adalah *American Sign Language*(ASL) (Chuan et al. 2014),(Oz & Leu 2007),(Marin et al. 2014),(Zaki & Shaheen 2011), *French Sign Language*(FSL), *Germany Sign Language*(GSL), dan *Arabic Sign Language*(ArSL) (Elons et al. 2013),(Mohandes et al. 2014),(Mohandes et al. 2015),(Elons et al. 2014).

Indonesia memiliki dua sistem bahasa isyarat nasional yaitu Berkenalan dengan Sistem Isyarat Indonesia (BISINDO) dan Sistem Isyarat Bahasa Indonesia (SIBI) (Sugianto & Samopa 2015). BISINDO telah dikembangkan oleh orang penyandang tuna rungu itu sendiri melalui Gerakan Kesejahteraan Tuna Rungu Indonesia (GERKATIN), sedangkan SIBI dikembangkan oleh orang normal yang mengadopsi bahasa isyarat ASL.

Orang penyandang tunarungu dan tunawicara menerapkan bahasa isyarat BISINDO dan SIBI untuk berkomunikasi sehari-hari. SIBI merupakan salah satu metode bahasa yang digunakan untuk membantu komunikasi antara penyandang tunarungu dan tunawicara (Departemen Pendidikan Nasional Indonesia, 2002). System bahasa isyarat SIBI menggunakan isyarat tatanan tangan dan gerakan tangan. Pada bahasa isyarat SIBI terdapat 26 huruf (24 huruf merupakan isyarat tangan yang berbentuk statis dan 2 huruf merupakan isyarat tangan yang berbentuk dinamis seperti huruf J dan Z) dan 10 isyarat angka (angka 0 sampai dengan angka 9).

2.2 Leap Motion Controller (LMC)

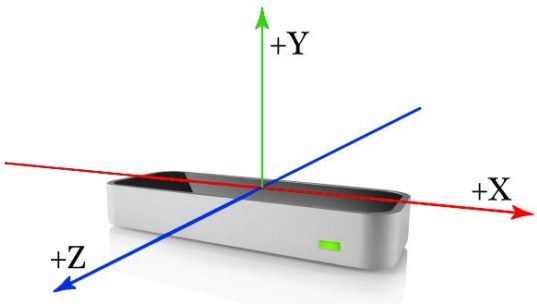
Sistem LMC merupakan sistem yang dapat mengenal gerakan tangan, jari tangan dan alat-alat yang memiliki bentuk seperti jari (Leap Motion 2014), (Leap Motion 2015). Perangkat ini beroperasi pada jarak yang dekat dengan presisi yang tinggi dengan frame tingkat pelacakan, laporan posisi yang diskrit, Gesture dan gerakan. LMC menggunakan sensor optik dan cahaya inframerah. Sensor yang diarahkan sepanjang arah sumbu *y* saat *controller* dalam pengoperasian standar dan memiliki bidang pandang 150 derajat. Jarak efektif LMC memanjang diatas perangkat dari 25 – 600 *milimeter* (1 inci – 2 kaki).

Proses deteksi dan pelacakan bekerja dengan baik ketika controller memiliki pandangan yang jelas, pandangan kontras yang tinggi dari suatu objek sepertiGambar 2.1. Perangkat LMC menggabungkan data sensor dengan model internal dari tangan manusia untuk membantu menangani kondisi pelacakan yang sulit.

Gambar 2.1Pandangan LMC terhadap tangan (Leap Motion 2015).

2.2.1 Sistem Koordinat

Sistem LMC menggunakan sistem koordinat *cartesian* pada tangan kanan seperti padaGambar 2.2. Titik pusat 0 berada pada tengah atas LMC. Sumbu *x* dan *z* terletak pada bidang horizontal, sumbu *x* berjalan sejajar dengan sisi panjang perangkat *LMC* dan sumbu *z* berada pada persilangan dengan sumbu *x*. Sumbu *y* berada pada bidang vertikal terhadap perangkat LMC, berbeda dengan sumbu *x* dan *z*, sumbu y hanya memiliki nilai positif yaitu berjalan dengan mengarah keatas.



Gambar 2.2Sistem koordinat *leap motion*.

Aplikasi LMC memiliki ukuran jarak, waktu, kecepatan dan sudut. Adapun satuan ukuran pada sistem LMC adalah seperti padaTabel 2.1.

Tabel 2.1 Satuan ukuran LMC

Ukuran	**Satuan**
Jarak	*Milimeter*
Waktu	*Microsecond* (kecuali dinyatakan lain)
Kecepatan	*Millimeter* / *second*
Sudut	*Radian*

2.2.2 *Motion Tracking Data*

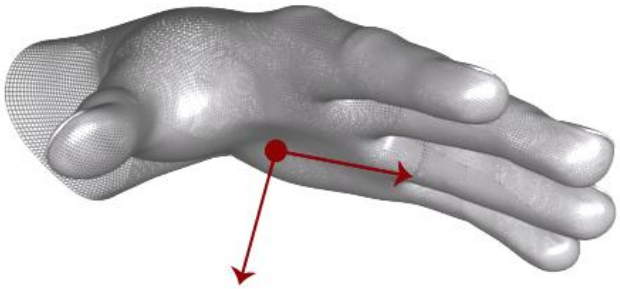
LMC sebagai sistem yang dapat mendeteksi tangan, jari dan alat yang berada pada bidang pandang LMC, memberikan peribahan data pada sebuah data atau frame data. Setiap objek *frame* menunjukkan *frame* yang mengandung daftar data deteksi seperti tangan, jari dan alat yang berbentuk seperti jari, serta deteksi gerakan dan factor yang menggambarkan gerakan selama waktu proses deteksi. Objekframemerupakan akar dari model data LMC. Pada *motion tracking data* terdapat beberapa objek yaitu *hands, arms, fingers, tools, gestures* dan *motions*.

2.2.2.1 *Hands*

Model tangan memberikan informasi tentang identitas, posisi, dan karakteristik lain dari tangan yang terdeteksi, lengan merupakan tempat dimana tangan terpasang, dan daftar jari yang terkait dengan tangan. Perangkat lunak LMC menggunakan model internal tangan manusia untuk memberikan pelacakan

yang prediktif bahkan ketika bagian dari tangan yang tidak terlihat. Model tangan selalu memberikan posisi lima jari, sehingga pelacakan optimal ketika tangan dan semua jari yang jelas terlihat. Perangkat lunak ini menggunakan bagian terlihat dari sisi model internal, dan observasi masa lalu untuk menghitung posisi yang paling mungkin dari bagian-bagian yang tidak terlihat saat pelacakan baru. Perhatikan bahwa gerakan halus jari terselip terhadap tangan atau terlindung dari sensor LMC biasanya tidak terdeteksi. *Hand confidence* adalah sebuah fungsi yang menunjukkan seberapa baik data yang diamati sesuai dengan model internal.

LMC dapat mendeteksi lebih dari dua tangan ketika ada tangan orang lain yang muncul, namun *frame* hanya dapat mendeteksi dua tangan yaitu tangan kanan dan tangan kiri. Untuk hasil pelacakan yang optimal harus menggunakan maksimal dua tangan. Pada LMC terdapat posisi *palm* normal dan vector arah yang menentukan orientasi tangan seperti pada Gambar 2.3.



Gambar 2.3*Palm* normal dan arah vector.

2.2.2.2 *Arms*

Arms (lengan tangan) adalah objek tulang seperti yang menyediakan titik orientasi, panjang, lebar, dan ujung lengan. Ketika siku tidak dalam pandangan, LMC memperkirakan posisi koordinatnyaberdasarkan pengamatan sebelumnya.

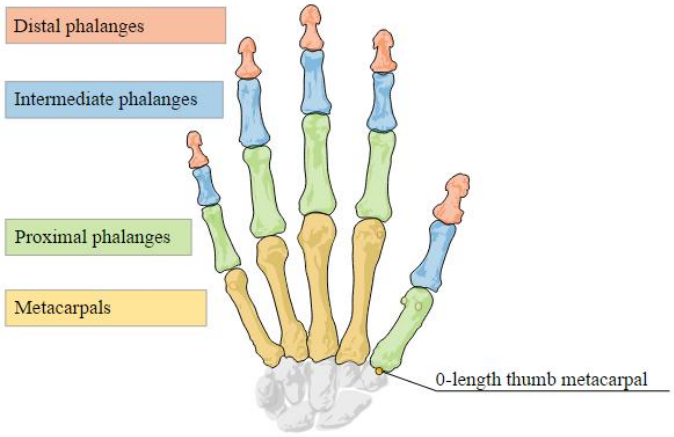
2.2.2.3 *Fingers*

LMC memberikan informasi tentang masing-masing jari pada tangan. Jika semua atau sebagian dari jari tidak terlihat, karakteristik jari diperkirakan berdasarkan pengamatan sebelumnya dan model anatomi tangan. Jari diidentifikasi dengan tipe nama yaitu: *thumb*(jari jempol), *index*(jari telunjuk), *middle*(jari tengah), *ring*(jari manis), and *pinky*(jari kelingking). Posisi

finger tip (ujung jari) dan arah vector memberikan posisi koordinat ujung jaridan arah jari menunjuk sepertiGambar 2.4.

Gambar 2.4Posisi ujung jari dan arah vektor.

Sebuah objek jarimenyediakan objek tulang yang menggambarkan posisi koordinat dan orientasi masing-masing tulang jari. Setiap jariterdapat empat titik koordinat tulang dari pangkal jari sampai ke ujung jari seperti padaGambar 2.5.



Gambar 2.5Tulang tulang pada jari tangan.

Tulang-tulang pada tangandapat diidentifikasikan sebagai berikut:

1. *Metacarpal* - tulang di dalam tangannya menghubungkan jari ke pergelangan tangan (kecuali ibu jari).
2. *Proksimal Phalanx* - tulang di dasar jari, terhubung ke telapak tangan.
3. *Intermediate Phalanx* - tulang tengah jari, antara ujung dan pangkal.
4. *Distal Phalanx* - tulang terminal di ujung jari.

Pada model ini untuk jari jempol tidak cocok dengan sistem penamaan anatomi standar. Sebuah jari jempol dalam kenyataannya memiliki tiga tulang

yang berbeda dengan jari-jari lainnya yang memiliki empat tulang. Namun, untuk kemudahan proses pemrograman, pada jari jempol, sistemLMC mendefinisikan tulang *metacarpal* dengan panjang nol, sehinggajari jempol memiliki jumlah yang sama dengan tulang pada indeks yang sama dengan jari-jari lainnya. Akhirnya tulang metacarpal pada jari jempol diberi label *phalanx proximal* dan *phalanx proximal* diberi label sebagai *phalanx* intermediate.

2.2.2.4 *Tools*

Sebuah *tools* adalah sebuah objek seperti sebuah pensil yang ada pada tangan seperti padaGambar 2.6. *Tools*akan dideteksi oleh *leap motion*ketika memiliki bentuk seperti pensil. Definisi *tools* adalah sebuah alat yang berbentuk panjang, tipis dan lurus (Leap Motion 2015).

Gambar 2.6*Tools*.

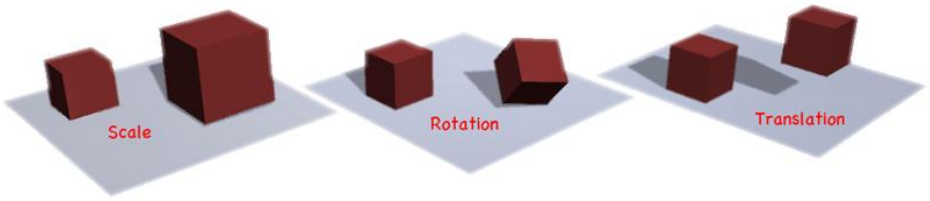
2.2.2.5 *Gestures*

Perangkat lunak *Leap Motion*mengenal pola pergerakan tertentu sebagai gerakan yang bisa menunjukkan maksud atau perintah pengguna. *Gestures* diamati untuk setiap *finger*atau alat individual. Laporan *software* LMC gerakan diamati dalam *frame*dengan cara yang sama bahwa laporan data pelacakan gerak lainnya seperti jari dan tangan.

2.2.2.6 *Motions*

Motions (gerakan) merupakan perkiraan dari tipe dasar gerakan yang melekat dalam perubahan tangan pengguna selama periode waktu. Gerakan termasuk skala, rotasi, dan translasi (perubahan posisi) seperti pada Gambar 2.7dan definisi *motions* dapat dilihat pada Tabel 2.2. Gerakan dihitung antara dua

frame yang bisa mendapatkan faktor gerak untuk adegan secara keseluruhan dariobjek*frame*. *Motions*juga bisa mendapatkan faktor yang terkait dengan satu tangan dari objek tangan(Leap Motion 2015).



Gambar 2.7*Motions*.

Motions dapat menggunakan faktor gerak dilaporkan untuk merancang interaksi dalam aplikasi. Misalnya, melacak perubahan posisi jari individu di beberapa frame data, gerakan bisa menggunakan faktor skala dihitung antara dua frame untuk membiarkan pengguna mengubah ukuran objek.

Tabel 2.2Tipe *motion*

Tipe *motion*	***Frame***	***Hand***
Skala	*Frame* skala menggambarkan gerak adegan benda menuju atau jauh dari satu sama lain. Misalnya, satu sisi bergerak lebih dekat dengan yang lain.	skala tangan menggambarkan perubahan jari menyebar.
Rotasi	*Frame* rotasi menggambarkan pergerakan diferensial benda dalam adegan. Misalnya, satu tangan ke atas dan yang lain ke bawah.	rotasi tangan menggambarkan perubahan dalam orientasi satu tangan.
Translasi	*Frame translation* menggambarkan rata-rata perubahan posisi semua objek di pemandangan. Misalnya, kedua tangan bergerak ke kiri, ke atas, atau ke depan.	*translasi* tangan menggambarkan perubahan posisi tangan.

2.3 Fitur Statis

Pada penelitian sebelumnya (Chuan et al. 2014) telah digunakan beberapa fitur statis untuk pengenalan bahasa isyarat ASL. Proses pembentukan fitur statis dilakukan berdasarkan titik-titik koordinat tangan seperti padaGambar 2.8.Fitur

statis yang digunakan adalah seperti fitur *pinch strength, grab strength, average distance, average spread, average tri-spread, extended distance, dip-tip projection, orderX,angle* :

a. *Average distance* adalah pengukuran jumlah jarak antara ujung setiap jari *tip* pada frame yang berdekatan. Asumsikan tip_t^n yang menunjukkan posisi *tip* dari jari *n* pada frame *t*, *n* = {1, 2, 3, 4, 5} untuk ibu jari (*thumb*), jari telunjuk (*index finger*), jari tengah (*middle finger*), jari manis (*ring finger*) dan jari kelingking (*pinky finger*). Adapun persamaan untuk *average distance* ditunjukkan pada Persamaan 2.1.

$$averageDistance = \frac{1}{T-1}\sum_{t=1}^{T-1}\sum_{n=1}^{5}|tip_{t+1}^{n} - tip_t^n| \quad (2.1)$$

dimana T adalah total jumlah *frame*dari tangan yang diambil dari *LMC* dan $|tip_{t+1}^{n} - tip_t^n|$ adalah jarak antara *tip* jari pada dua *frame*yang berdekatan.

b. *Average spread* adalah perhitungan jarak antara dua *tip* yang berdekatan pada setiap frame. $|tip_t^{n+1} - tip_t^n|$ adalah jarak antara dua *tip* yang berdekatan. Adapun persamaan untuk *average spread* ditunjukkan pada Persamaan 2.2.

$$averageSpread = \frac{1}{T}\sum_{t=1}^{T}\sum_{n=1}^{4}|tip_t^{n+1} - tip_t^n| \quad (2.2)$$

*Average tri-spread*adalah area segitiga (*triArea*) antara dua *tip* yang berdekatan dan titik tengah (*midpoint*) antara dua posisi *metacarpal* (*mcp* dapat dilihat pada

c. Gambar 2.8). Asumsikan tip_n , tip_{n+1} , dan $mcp_{n,n+1}$ merupakan titik koordinat 3D dari jari $n, n+1$ dan titik tengah dari kedua *metacarpal* pada kedua jari yang berdekatan. Area dari segitiga didefinisikan dengan tiga titik yang dihitung setengah dari *cross product* dari dua vektor $\overrightarrow{tip_n mcp_{n,n+1}}$ dan $\overrightarrow{tip_{n+1} mcp_{n,n+1}}$. Adapun perhitungan area segitigaditunjukkan pada Persamaan 2.3.

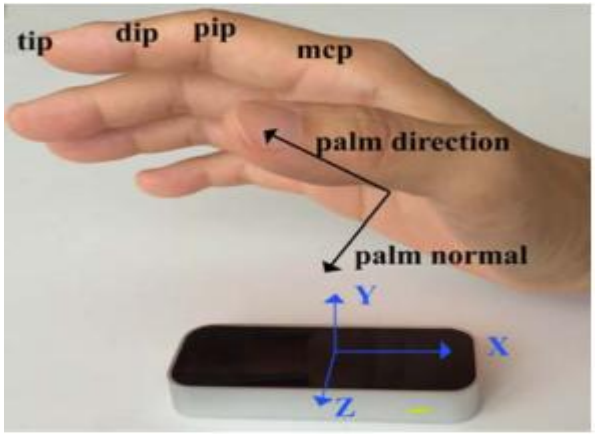
$$triArea_{n,n+1} = \frac{1}{2} \overrightarrow{tip_n mcp_{n,n+1}} \; x \; \overrightarrow{tip_{n+1} mcp_{n,n+1}}$$
$$= \frac{1}{2} \overrightarrow{|tip_n mcp_{n,n+1}||tip_{n+1} mcp_{n,n+1}|} \; sin(\theta) \quad (2.3)$$

dimana $\boldsymbol{\theta}$ adalah sudut antara dua vektor.

Average tri-spread dapat dihitung dengan menambahkan area segitiga dari semua pasangan jari dan dibagi dengan total dari jumlah frame. Adapun persamaan *average spread* ditunjukkan pada Persamaan 2.4.

$$averageTrispread = \frac{1}{T} \sum_{t=1}^{T} \sum_{n=1}^{4} triArea_{n,n+1} \quad (2.4)$$

d. *Extended distance* adalah jarak maksimum dari semua titik jari (*tip, dip, pip* dan *mcp* ditunjukkan padaGambar 2.8terhadap titik *palm center* (titik tengah tangan). Hasil nilai pada fitur ini memiliki satuan mm (*millimeter*).
e. *Dip-tip projection*adalah proyeksi dari vektor *tip* ke *dip* diatas vektor *palm normal*. Hasil nilai dari fitur ini memiliki satuan vektor.
f. *OrderX* adalah urutan jari sepanjang bidang *x-z* sehubungan dengan jari-jari lainnya.
g. *Angle* adalah sudut antara vektor arah jari dan bidang *x-z*.



Gambar 2.8 Capture tangan menggunakan LMC.

2.4 Fitur Dinamis

Fitur dinamis merupakan fitur gerak atau fitur yang bersifat dinamis yang didapatkan dari deteksi LMC terhadap tangan. Fitur dinamis yang digunakan pada metode ini adalah fitur gerakan tangan atau disebut dengan *hand gesture*(Chen et al. 2015).

Pembentukan fitur yang baik akan meningkatkan akurasi dalam pengenalan. Lintasan gerakan diproyeksikan kedalam bidang *X0Z* yang merupakan prinsip pada bidang. Untuk setiap posisi titik $palmP_t\ (x_t,\ y_t,\ z_t)$ dalam urutan gerakan dinamis, orientasinya dalam bidang lintasan dapat digambarkan oleh vektor$\xrightarrow[P_{t-1}\quad P_t]{}$ orientasi sudut mutlak dapat dicatat $\alpha_t \in (0, 360°)$ dapat dilihat pada Persamaan (2.5) (2.6) dan (2.7).

$$\Delta z = z_t - z_{t-1} \tag{2.5}$$

$$\Delta x = x_t - x_{t-1} \tag{2.6}$$

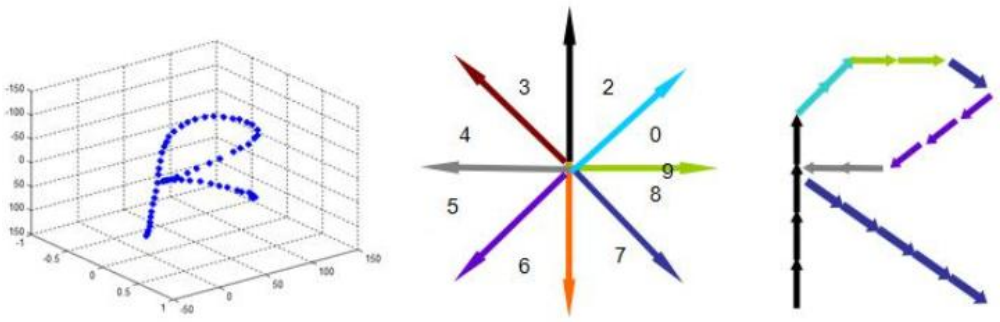
$$\alpha_t = \begin{cases} \arctan\left(\frac{\Delta z}{\Delta x}\right) * \left(\frac{180}{\pi}\right) + 180, & jika\ \Delta x < 0, \\ \arctan\left(\frac{\Delta z}{\Delta x}\right) * \left(\frac{180}{\pi}\right) + 360, & jika\ \Delta z < 0, \\ \arctan\left(\frac{\Delta z}{\Delta x}\right) * \left(\frac{180}{\pi}\right), & jika\ \Delta x > 0\ dan\ \Delta z \geq 0, \end{cases} \tag{2.7}$$

dimana :

Δz = nilai selisih antara dua vector *z*,

Δx = nilai selisih antara dua vector *x*.

Setelah mencapai urutan orientasi sudut, untuk meningkatkan tingkat pengenalan, sudutα_t dikuantisasi dengan membaginya dengan 45° dan diberikan kode dari 1 sampai 9 (9 merepresentasikan 0°). Seperti ditunjukkan Gambar 2.9(A) adalah contoh pergerakan dalam bidang *X0Z*, sedangkan Gambar 2.9(B) adalah acuan kuantisasi dari orientasi 360^{o}, dan Gambar 2.9(C) adalah contoh hasil kuantisasi untuk huruf R.



A B C

Gambar 2.9Proses *quantization* (A) Contoh pergerakan dalam bidang *XOZ*, (B) Acuan *quantization*, (C) Hasil *quantization.*

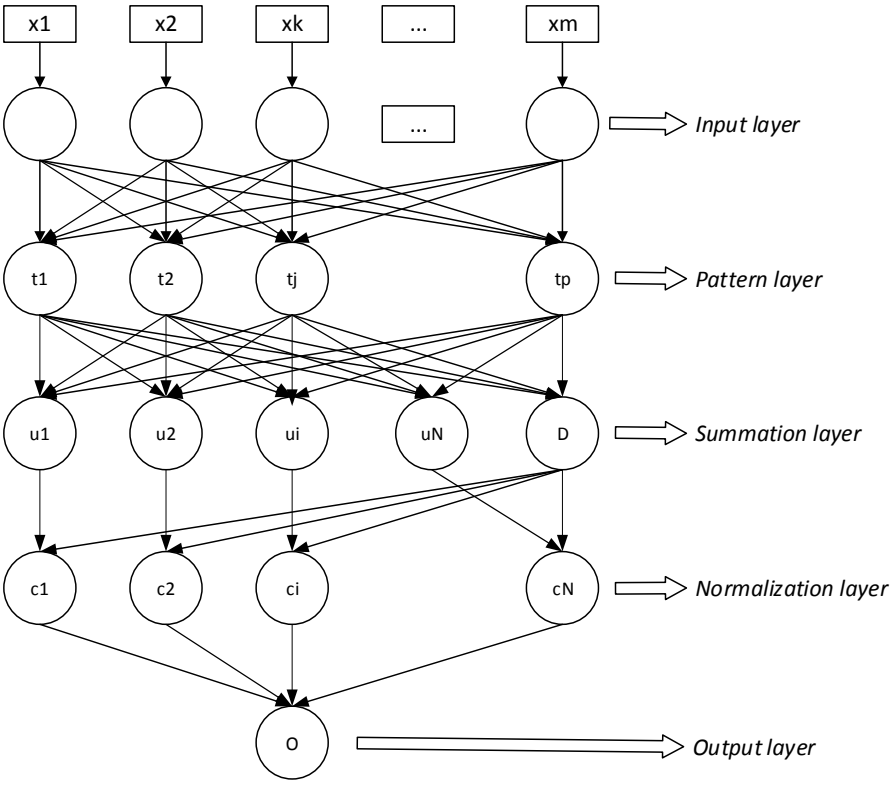
2.5 *Decission Tree*

Decission tree atau disebut juga pohon keputusan merupakan pohon yang dapat digunakan sebagai penalaran untuk menyelesaikan dan memerikan jawaban dari masalah yang dimasukkan. Pohon yang dibentuk tidak selalu berbentuk biner. Jika semua fitur dalam dataset *menggunakan* dua jenis nilai kategorikal maka bentuk pohon yang didapatkan berupa pohon biner. Jika dalam fitur berisi lebih dari dua kategorikal atau menggunakan tipe numerikmaka bentuk pohon yang didapatkan biasanya tidak berupa pohon biner (Gorunescu 2015).

2.6 Logarithmic Learning for Generalized Classifier Neural Network (L-GCNN)

L-GCNN tidak seperti jaringan syaraf tiruan berbasis radial basis function lainnya.Arsitektur L-GCNN dapat dilihat padaGambar 2.10.L-GCNN menggunakan fungsi *logarithmic* sehingga dapat mengurangi jumlah iterasi untuk mencapai *error* minimal. Metode ini memiliki 5 *layer* utama yaitu *input*, *pattern*, *summation*, *normalization* dan *outputlayer*.

Input layer mengirimkan vektor input *x* terpilih menuju ke *pattern layer*. *Pattern layer* terdiri atas satu *neuron* untuk tiap training datum. *Neuron* pada *patternlayer* digunakan untuk menghitung jarak *euclidian* antara *vectorinputx* dan *trainingdatavectort* dengan menggunakan Persamaan 2.8. Dengan *P* menunjukan jumlah dari data taining.



Gambar 2.10Arsitektur L-GCNN.

$$dist(j) = \left\| x - t_j \right\|^2 , 1 \leq j \leq P \tag{2.8}$$

Dengan :

x = *Vectorinputx,*
t = *Trainingdatavector* t,
P = Menunjukan jumlah dari *trainingdata*.

Output pada*pattern layer* ditentukan dengan menggunakan fungsi aktivasi RBF. Seperti yang ditunjukkan pada Persamaan 2.9.

$$r(j) = e^{-1*\frac{dist\ (j)}{2\sigma^2}}, \qquad 1 \leq j \leq P \tag{2.9}$$

Dengan
$dist(j)$ = Jarak *Euclidean,*
$2\sigma^2$ = Varians data,
P = Jumlah total datatraining.

Sebagai metodologi L-GCNN yang berbasis regresi, maka L-GCNN dibuat dengan struktur *one vs all discriminative*. Oleh karena itu, tiap *training datum*

memiliki N nilai yang ditentukan dengan menentukan apakah data tersebut termasuk kedalam suatu kelas atau tidak. Jika *training datum* termasuk kedalam *ith class* maka *class* ke *ith* tersebut akan bernilai 0.9 dan *class* yang lainnya bernilai 0.1.

$$y(j,i) = \begin{cases} 0.9, & t_j\ termsuk\ kelas\ ith\ 1 \le i \le N \\ 0.1. & else\ 1 \le j \le p \end{cases} \quad (2.10)$$

Summation Layer memiliki *N*+1 *neuron* dimana *N* adalah jumlah keseluruhan *class* yang ada dan 1 merupakan neuron *denominator*. Pada *summation layer*, GCNN menggunakan *diverge effect* term pada N *neuron* untuk kinerja pengklasifikasian yang lebih baik. *Diverge effect term* menggunakan bentuk eksponensial dari $y(j,i) - ymax$, Persamaan 2.11 untuk meningkatkan efek dari $y(j,i)$. Maksud digunakannya fungsi eksponensial adalah untuk menyediakan titik temu (*convergence*) dengan *minimal error* antara *limit*. *Diverge effect term* memberikan dua keuntungan utama terhadap GCNN. Dengan meningkatkan efek dari $y(j,i)$, data akan termasuk kedalam salah satu kelas, dan terpisah satu sama lain. Dengan mendapatkan keuntungan dari fungsi eksponensial, masalah *overfitting*, pendekatan umum *gradient descent* dapat ditekan.

$$d(j,i) = e^{(y(j,i)-ymax)} * y(j,i) \quad (2.11)$$

Dimana $d(j,i)$ menunjukkan *diverge effect term* dari *jth training data* dan *ithclass*. y_{max} diinisialisasikan dengan dengan nilai 0.9 yang menunjukkan nilai maksimum dari $y(j,i)$ dan diupdate dengan nilai maksimum dari output layer untuk tiap iterasi.

Pada layer ini, N *neuron*menghitung jumlah dari dot produk dari *diverge effect term* dan *pattern layer outputs* seperti yang diberikan pada Persamaan 2.12.

$$u_i = \sum_{j=1}^{p} d(j,i) * (r(j), 1 \le i \le N \quad (2.12)$$

Dengan :
$d(j,i)$ = *Diverge effect,*
$r(j)$ = *Output pattern layer,*

N = Banyaknya *neuron*.

Kemudian *neuron* yang lainnya menghitung *denominator* seperti yang ditunjukkan pada Persamaan 2.13.

$$D = \sum_{j=1}^{p} r(j) \tag{2.13}$$

Dengan :
$r(j)$ = *Output pattern layer*.

Pada *Normalization layer* , terdapat N *neuron* yang ditampilkan, tiap *class* dan *output* dari *neurontersebut* dihitung dengan menggunakan Persamaan 2.14.

$$c_i = \frac{u_i}{D}, 1 \leq i \leq N \tag{2.14}$$

Dengan :
u_i = *Output summation layer*,
D =*Output neuron denominator*,
N = banyaknya *neuron*.

Kemudian tahap akhir, pada lapisan terakhir mekanisme keputusan pemenang diberikan *menggunakan* Persamaan 2.15 dengan memilih nilai maksimum dari *output* dari *normalization layer*.

$$[o, id] = \max(c) \tag{2.15}$$

Dengan :
o = nilai,
id = kelas,
Max(c) = nilai *outputneuron* pada *layernormalization* tertinggi.

L-GCNN menggunakan metode pembelajaran *logarithmiccostfunction* untuk pengoptimalisasian *smoothing parameter*. Tidak seperti GCNN yang biasanya, *cost function* didefenisikan dengan menggunakan fungsi *logarithmic* bukan *error* kuadrat seperti yang ditunjukkan pada Persamaan 2.16, dimana e menunjukkan *cost function*, $y(z, id)$ menunjukkan nilai dari data *training input* ke zth untuk $idth$ *class dan* c_{id} merupakan nilai dari *class* pemenang. Ide dibelakang metode yang diajukan ini adalah dengan tujuan untuk memaksimalkan kemungkinan yang ada.

Hasil berbagai operasi L-GCNN dan fungsi aktifasi RBF, turunan dari *cost function* didefinisikan dan berkelanjutan. Oleh karena itu, tanpa solusi numerik apapun metode yang diusulkan dapat digunakan untuk melakukan pelatihan terhadap GCNN.

$$e = (y(z,id) * \log(c_{id})) + \big(1 - y(z,id)\big) * \log(1 - c_{id}) \tag{2.16}$$

Dengan :
$y(z,id)$ = nilai dari data *training input* ke zth untuk $idthclass$,
$\log(c_{id})$ = nilai dari *class* pemenang.

Saat bagian pertama dari Persamaan 2.16 menunjukkan kuantitas *convergence* dengan nilai target sebesar 0.9 jika datum termasuk kedalam salah satu *class* kemudian bagian kedua dari persamaanmenunjukkankwantitas *convergence* sebesar 0,1 jika datum tidak termasuk kedalam *class* tersebut. Oleh karena itu, jika nilai target adalah 0.9 maka nilai biaya adalah ditentukan sebagian besar oleh bagian pertama dari Persamaan 2.16 dan jika nilai target adalah 0.1 maka bagian kedua dari *cost function* akan sangat berpengaruh.

Smoothing parameter diperbaharui degan menggunakan *gradient* dari fungsi biaya (e) dengan menggunakan Persamaan (2.17-2.21), dimana lr adalah *learning rate*.

$$\sigma_{new} = \sigma_{old} + lr * \frac{\partial e}{\partial \sigma} \tag{2.17}$$

$$\frac{\partial e}{\partial \sigma} = y(z,id) * \left(\frac{\frac{\partial c_{id}}{\partial \sigma}}{c_{id}}\right) + (1 - y(z,id) * \left(\frac{\frac{\partial c_{id}}{\partial \sigma}}{c_{id}}\right) \tag{2.18}$$

$$\frac{\partial c_{id}}{\partial \sigma} = \frac{b(id) - l(id) * c_{id}}{D} \tag{2.19}$$

$$b(id) = 2 * \sum_{j=1}^{p} d(j,i) * (r(j), * \frac{dist(j)}{\sigma^3} \tag{2.20}$$

$$l(id) = 2 * \sum_{j=1}^{p} (r(j), * \frac{dist(j)}{\sigma^3} \quad (2.21)$$

Dengan :

lr	=	*learningrate*,
$dist(j)$	=	jarak *euclidean*,
$d(j,i)$	=	*Diverge effect*,
$r(j)$	=	*Outputpatternlayer*,
$y(z,id)$	=	nilai dari data *training input* ke zth untuk $idthclass$,
c_{id}	=	nilai dari *class* pemenang.

L-GCNN menggunakan pembelajaran secara online untuk meminimalisir total biaya (*cost*). Selama c_{id} memiliki nilai dengan rentang antara 0.1-0.9 dan *logarithm* dari c_{id} berada pada rentang (-2, -0,3), walaupun sampel pelatihan terklasifikasi dengan benar, *smoothing parameter* akan tetap diperbaharui. *Gradient* dari *cost function* hanya menunjukkan kuantitas perubahan. Semakin *convergence* maka semakin sedikit perubahan yang diterapkan pada *smoothing parameter*. Pelatihan berakhir jika jumlah iterasi atau toleransi kesalahan melebihi batas. Selama *smoothing parameter* menunjukkan lebar dari fungsi Gaussian, maka nilainya harus bernilai positif. L-GCNN memeriksan nilai baru dari *smoothing parameter* berlawanan dengan *contraint*nya sebelum memperbaruinya. Jika berkuran dibawah nol maka *smoothing parameter* tidak akan *diupdate.*

Algoritma dari langkah-langkah pelatihan L-GCNN dapat dilihat pada Algoritma 2.1 dimana *epoch* menunjukkan banyaknya jumlah iterasi yang digunakan pada algoritma pelatihan, te menunjukkan *errortotal* yang dihitung terhadap keseluruhan *training data* pada iterasi sebelumnya dan a_{te} menunjukkan *total error* yang diterima. Ketika salahsatu kriteria penghentian tercapai, maka pelatihan dihentikan dan *smoothing parameter* yang dapat diterima sebagai nilai *smoothing parameter* yang optimal dibawah toleransi kesalahan dan jumlah maksimum iterasi dicapai.

Algoritma 2.1. L-GCNN untuk proses pelatihan :
Input : epoch, lr, data training, a_{te}
Output : parameter smoothing
Inisialisasi parameter smoothing σ dan y_{max}
while iterasi ≤ epoch
for setiap data uji coba t_j
if iterasi > 1
if$\left(\sigma_j + lr * \frac{\partial e}{\partial \sigma_{id}}\right) > 0$
Update σ_j dengan $\frac{\partial e}{\partial \sigma}$
Temukan jarak euclidean distance antara data input dan data training, dist (j)
Lakukan fungsi aktivasi RBF, r (j)
for setiap kelas ; i
Hitung divergen effect term,
$d\ (j,\ i) =\ e^{(y(j,i)-(y_{max}\ )} * y\ (j,i)$
Hitung $u_i =\ \sum_{j=1}^{p} d(j,i) * r\ (j)$ dan $D =\ \sum_{j=1}^{p} r\ (j)$
Hitung normalisasi nilai layer neuron
$c_i =\ \frac{u_i}{D}$
end for
Temukan pemenang neuron dan nilainya ; [o, id] = max (c)
Untuk update divergen effect term nilai neuron pemenang yang akan tersimpan ; cmax (iterasi) = c_{id}
Hitung logarithmic cost
$$e = (y\ (z,\ id) * \log(c_{id})) + \left(1 - y(z,\ id)\right) * \log(1 -\ c_{id})$$
Dimana z menunjukkan input ke-z
end for
y_{max} = max (cmax)
increment iterasi
if $te\ \leq\ a_{te}$
Stop training
end while

L-GCNN memiliki dua proses yaitu proses pelatihan dan proses uji coba. Pada proses pelatihan *input*-nya adalah *epoch*, *learning rate*, *data training*, dan *error minimum*, sedangkan outputnya adalah nilai *parameter smoothing.* Pada proses uji coba *input*-nya adalah *parameter smoothing* yang didapatkan dari proses pelatihan dan data uji coba, sedangkan *output* dari proses uji coba adalah kelas. Untuk proses pelatihan dapat dilihat pada Algoritma 2.1. sedangkan untuk proses uji coba dapat dilihat pada Algoritma 2.2.

Algoritma 2.2. L-GCNN untuk proses uji coba :

Input : parameter smoothing, data testing

Output : kelas

for setiap data training; t_j

Temukan euclidean distance data test dengan data training, dist(j)

Tentukan dengan fungsi aktifasi RBF, r(j)

for setiap kelas; i

hitung divergen effect term,

hitung jumlah dari dot produk dari diverge effect; U_i

dan denominator; D

hitung nilai neuron layer normalisasi; c_i

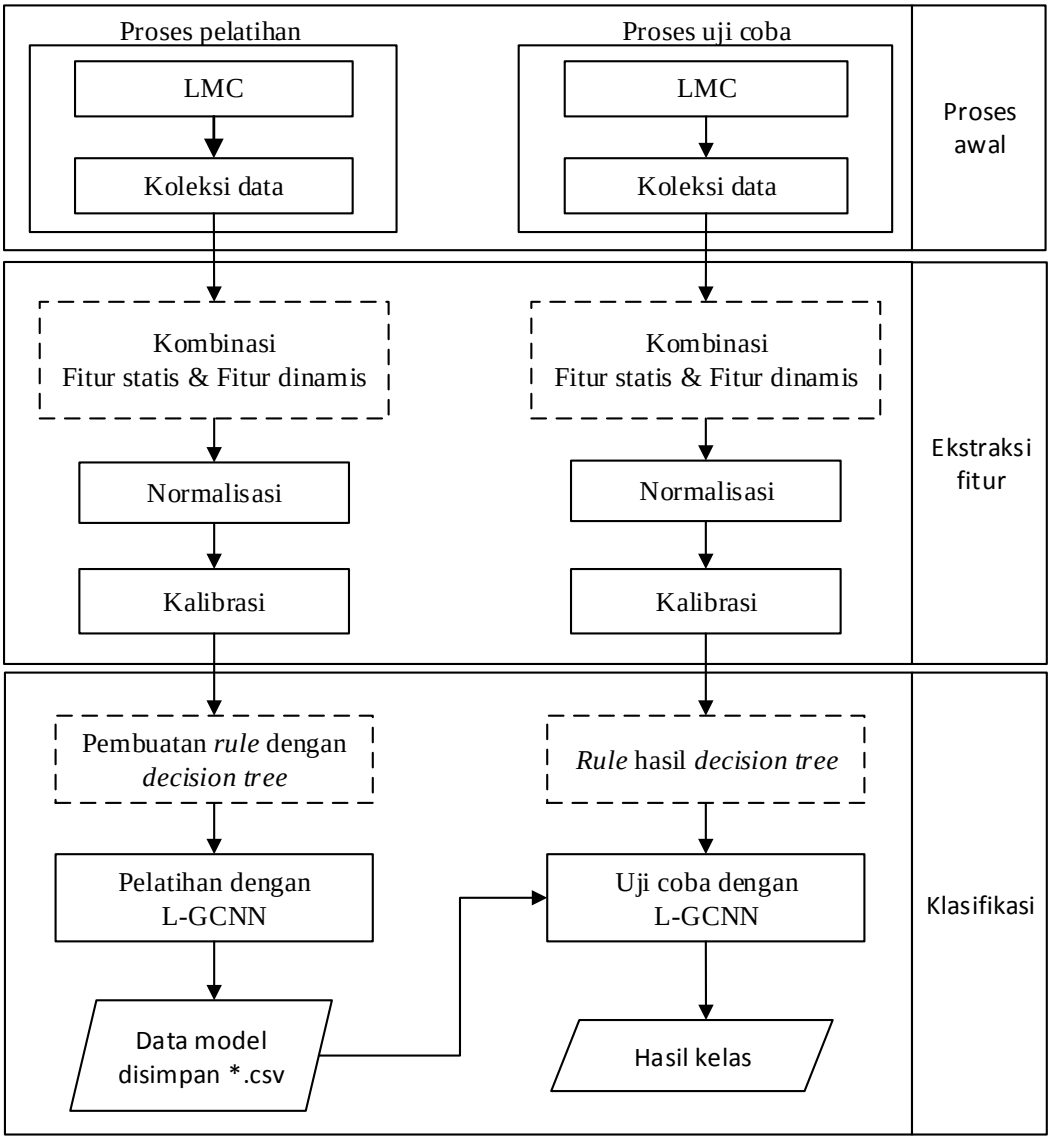
end for

Temukan neuron pemenang dan nilainya; [o, id] = max(c)

end for

BAB 3
METODOLOGI PENELITIAN

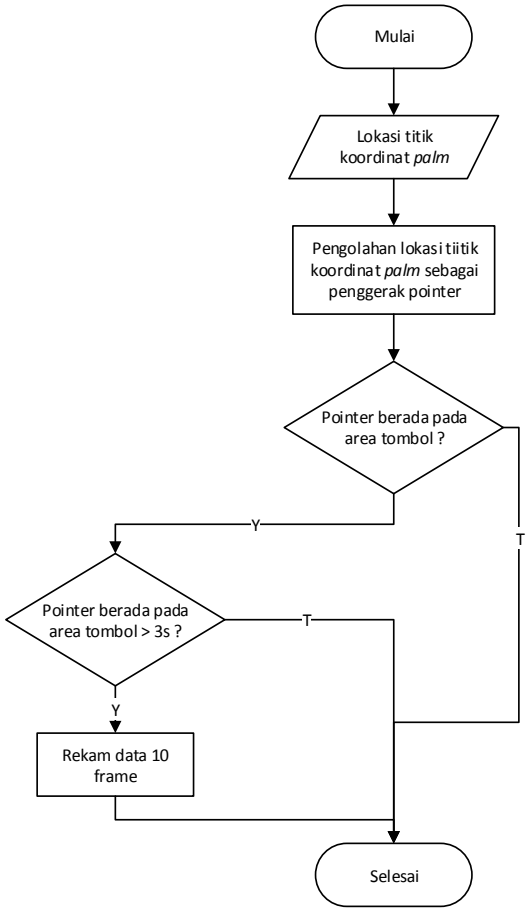
Metode dalam penelitian ini terdiri atas tiga tahapan utama yaitu: koleksi data dari LMC, ekstraksi fitur dan proses klasifikasi. Adapun ketiga tahapan utama penelitian dan bagian kontribusi ditunjukkan padaGambar 3.1.



Gambar 3.1Sistem pengenalan bahasa isyarat SIBI.

3.1 Proses Awal

Pada tahap *preprocessing* (proses awal) seperti pada Gambar 3.2adalah sebuah proses untuk mendapatkan koleksi data input yang didapatkan dari deteksi LMC terhadap tangan yang menunjukkan hasil titik-titik koordinat tangan dan gerakan tangan. Data titik-titik koordinat direkam sebanyak 10 *frame* untuk sekali perekaman atau setiap pembentukan fitur.Untuk sebuah huruf yang bersifat dinamis, huruf J membutuhkan gerakan yang lebih pendek daripada huruf Z. Pada penelitian ini digunakan sebanyak 10 *frame* karena jumlah tersebut dinilai tidak terlalu sedikit dan tidak terlalu banyak, sehingga 10 frame tersebut lebih tepat ketika digunakan untuk pergerakan dari sebuah huruf yang membutuhkan gerakan yang panjang maupun yang pendek. LMC dapat menangkap titik-titik koordinat dari lengan, tangan dan daftar jari yang berada pada area deteksi LMC ditunjukkan padaGambar 3.2.



Gambar 3.2 Tahap proses awal.

3.2 Ekstraksi Fitur

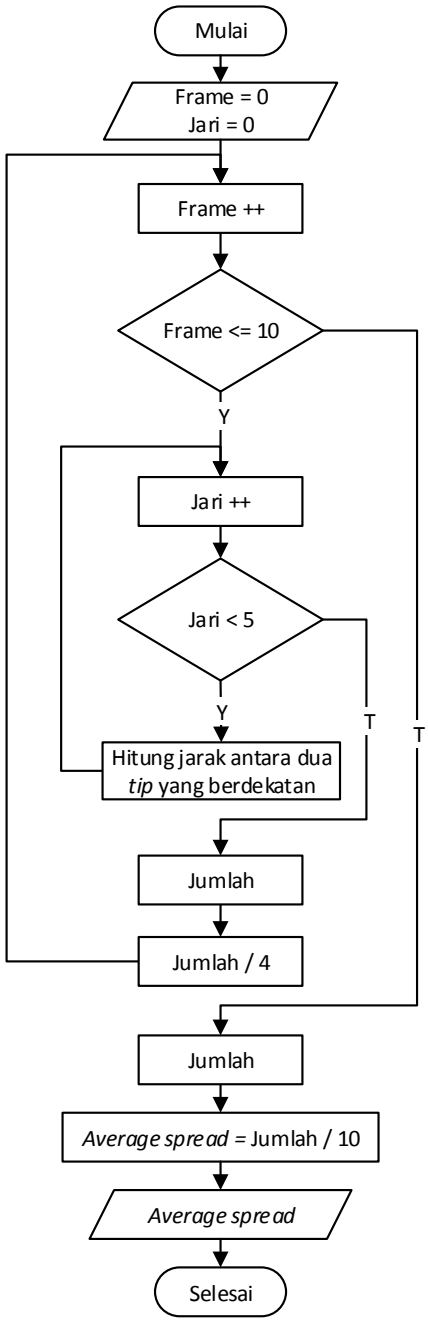
Input dari tahapan ini adalah hasil koleksi data titik-titik koordinat hasil dari proses tahap *preprocessing* sebelumnya. Ada dua macam fitur yang digunakan pada penelitian ini yaitu fitur statis dan fitur dinamis. Fitur statis adalah beberapa fitur yang bersifat statis yang didapatkan dari titik koordinat tangan pada LMC, sedangkan fitur dinamis adalah fitur yang bersifatdinamis yang didapatkan dari gerakan tangan pada LMC. Fitur statis yang digunakan pada penelitian ini adalah *average spread,average tri-spread*, dan *extended distance* sedangkan fitur dinamis yang digunakan adalah *hand gesture*. Adapun penjelasan dari fitur statis dan fitur dinamis dijelaskan pada subbab berikut.

3.2.1 Fitur Statis

Fitur statis adalah fitur yang dibentuk berdasarkan pada hasil dari titik-titik koordinat posisi tangan dalam keadaan diam atau posisi tangan yang tidak bergerak. Fitur statis dibentuk sebanyak sepuluh frame, setiap *frame* berisi titik-titik koordinat posisi tulang-tulang jari dan tangan dalam bidang 3D yaitu bidang XYZ. Pada fitur statis digunakan sebanyak 10 frame, selanjutnya untuk sepuluh *frame* akan diambil nilai rata-ratanya. Untuk membentuk fitur statis yang digunakan pada penelitian ini akan dibentuk beberapa fitur statis yaitu : fitur *average spread*, *average tri-spread* dan *extended distance*yang dijelaskan pada subbab berikut :

3.2.1.1 *Average Spread*

Average spread adalah nilai rata-rata dari sebaran jari atau nilai rata-rata dari perhitungan jarak antara dua *tip* (dua ujung jari) yang berdekatan pada setiap *frame*. $|tip_t^{n+1} - tip_t^n|$ adalah perhitungan jarak antara dua *tip* yang berdekatan. Dimana *t* adalah urutan *frame* dan *n* adalah urutan jari. Hasil *output* dari perhitungan *average spread* adalah berbentuk vektor. Adapun proses perhitungan *average spread* ditunjukkan padaGambar 3.3. Sedangkan perhitungan fitur lebih detail dapat dilihat pada Persamaan 2.2.



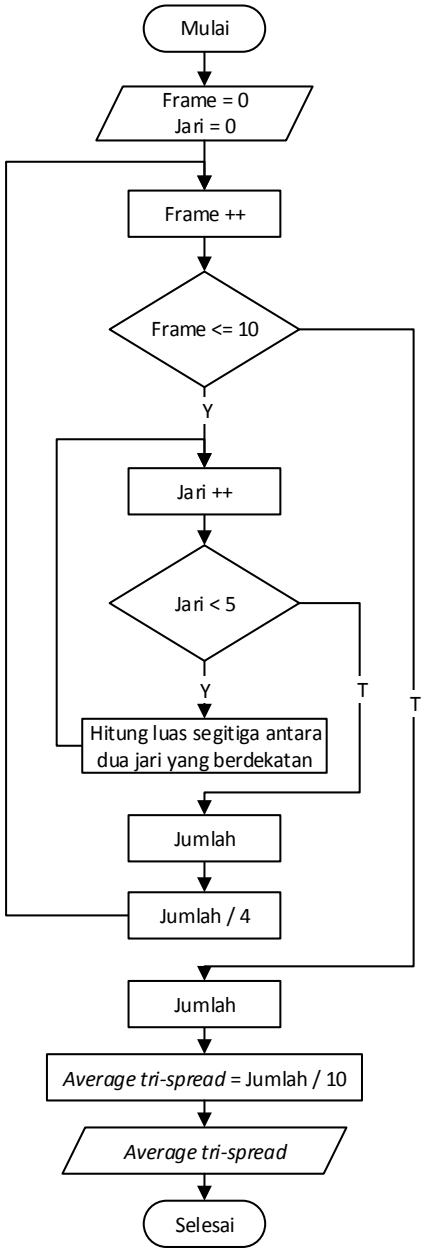
Gambar 3.3 Perhitungan fitur *average spread*

3.2.1.2 *Average Tri-Spread*

Average tri-spread adalah nilai rata-rata dari luas segitiga (*triArea*) antara dua *tip* yang berdekatan dan titik tengah (*midpoint*) antara dua posisi *metacarpal* (*mcp* dapat dilihat padaGambar 2.8). Asumsikan tip_n, tip_{n+1}, dan $mcp_{n,n+1}$ merupakan titik koordinat 3D dari jari $n, n+1$ dan titik tengah dari kedua *metacarpal* pada kedua jari yang berdekatan. *Area* dari segitiga didefinisikan

dengan tiga titik yang dihitung setengah dari *cross product* dari dua vektor $\overrightarrow{tip_n mcp_{n,n+1}}$ dan $\overrightarrow{tip_{n+1} mcp_{n,n+1}}$.

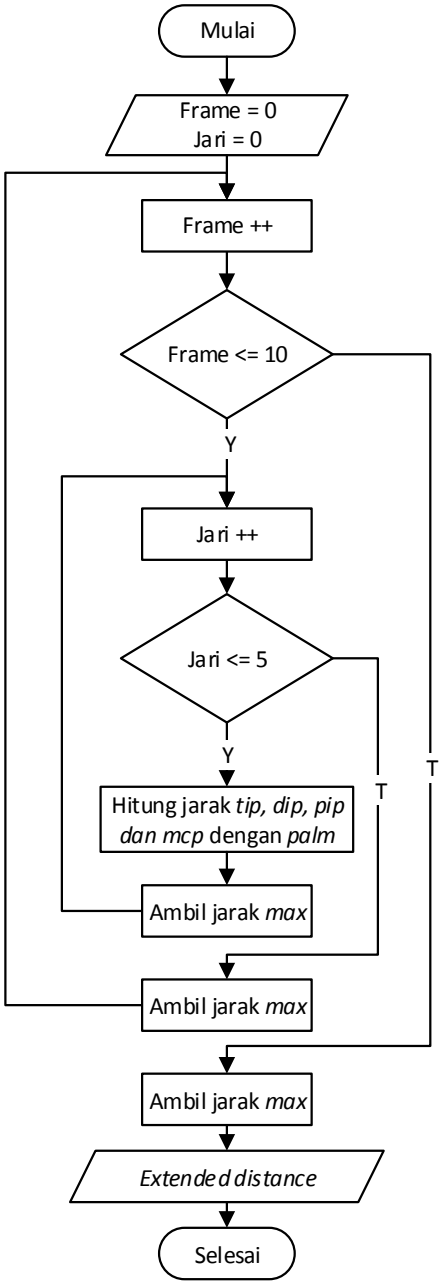
Adapun persamaan untuk perhitungan luas segitigaditunjukkan pada Persamaan 2.3.*Average tri-spread* dapat dihitung dengan menambahkan area segitiga dari semua pasangan jari dan dibagi dengan total dari jumlah *frame*. Adapun proses perhitungan *average tri-spread* ditunjukkan padaGambar 3.4.



Gambar 3.4 Perhitungan fitur *average tri-spread.*

3.2.1.3 *Extended Distance*

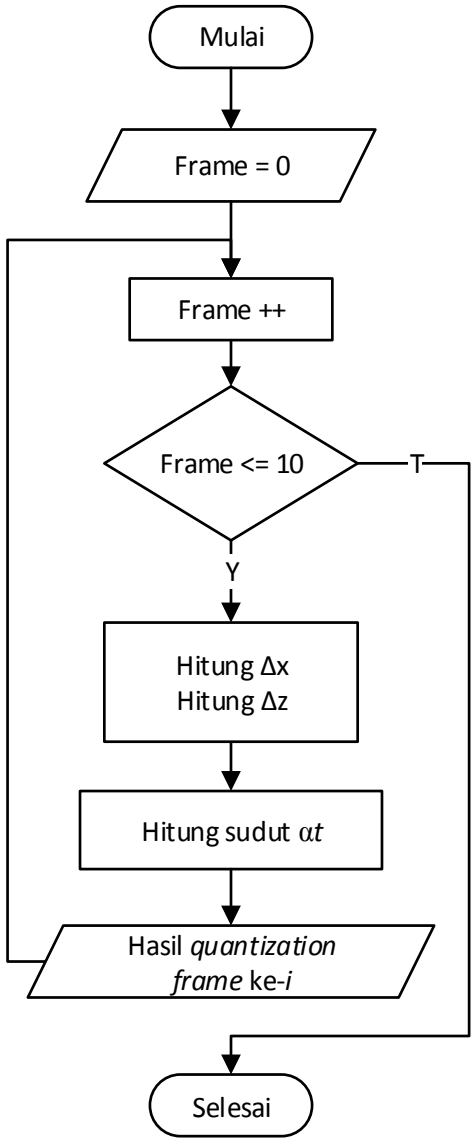
Extended distance adalah jarak maksimal dari semua titik jari (*tip, dip, pip* dan *mcp* ditunjukkan pada Gambar 2.8) terhadap titik *palm center* (titik tengah tangan). Hasil nilai*output* pada fitur ini adalah nilai maksimal dari semua jarak dan memiliki satuan mm (*millimeter*).



Gambar 3.5 Perhitungan fitur *extended distance*

3.2.2 Fitur Dinamis

Fitur dinamis merupakan fitur gerak atau fitur yang bersifat dinamis yang didapatkan dari deteksi LMC terhadap gerakan tangan. Fitur dinamis yang digunakan pada metode ini adalah fitur gerakan tangan atau disebut dengan *hand dynamic gesture.*Perhitungan fitur dinamis dapat dilihat pada Gambar 3.6



Gambar 3.6 Perhitungan fitur dinamis

Untuk mendapatkan fitur gerakan dinamis dari gerakan tangan digunakan lintasan gerakan yang diproyeksikan dalam bidang $X0Z$, sehingga Y dianggap nol.

Untuk setiap posisi titik *palm* $p_t(x_t, y_t, z_t)$ dalam urutan gerakan dinamis, orientasi dari bidang lintasan dapat digambarkan oleh vektor $\xrightarrow[p_{t-1}p_t]{}$, orientasi sudut dapat dinotasikan sebagai $\alpha_t \in (0, 360°)$ yang ditunjukkan pada Persamaan (2.5) (2.6) dan (2.7). Setelah mencapai urutan orientasi sudut, untuk meningkatkan tingkat pengenalan, sudut α_t dikuantisasi dengan membaginya dengan 45° dan kode dari 1 sampai 9 (9 merepresentasikan 0°) dapat dilihat padaGambar 2.9.

3.3 Penggabungan Fitur Statis dan Dinamis

Setelah dibentuk fitur statis dan fitur dinamis maka akan digabungkan antara fitur statis dengan fitur dinamis. Fitur statis yang didapatkan adalah sebanyak tiga fitur, sedangkan fitur dinamis yang didapatkan adalah sebanyak sembilan fitur. Fitur statis dan fitur dinamis akan digabung langsung dengan setiap kolom berisi satu fitur. Jumlah fitur setelah penggabungan adalah sebanyak 12 fitur, sehingga fitur terbentuk sebanyak 12 kolom. Fitur akan disimpan kedalam file *.csv yang akan digunakan sebagai dataset pengenalan bahasa isyarat. Adapun contoh kombinasi fitur statis dengan fitur dinamis dapat dilihat pada Tabel 3.1. Untuk data fitur yang digunakan pada penelitian ini dapat dilihat pada Lampiran 1.

Tabel 3.1 Contoh kombinasi fitur statis dan dinamis

Data ke-	Fitur statis			Fitur dinamis									Kelas
	Average spread	Average tri-spread	Extended distance	*Frame* ke-									
				1	2	3	4	5	6	7	8	9	
1	24.04	740.86	55.8	0	0	0	0	0	0	0	0	0	**A**
2	23.34	747.05	56.23	0	0	0	0	0	0	0	0	0	**A**
3	37.25	1328.68	217.35	0	0	0	3	3	4	4	4	0	**J**
4	34.75	1111.25	218.15	0	0	0	3	3	3	4	4	0	**J**

3.4 Normalisasi

Normalisasi data fitur (*feature scalling*) dilakukan untuk merubah data fitur kedalam *range* tertentu sehingga data fitur akan lebih proporsional. Nilai data fitur maksimum dan nilai data fitur minimum setiap data fitur didapatkan dari hasil

pengamatan yang dilakukan oleh penulis. Nilai fitur tangan setelah dinormalisasi X' dihasilkan dari nilai data fitur sebelum dinormalisasi X yang dikurang dengan nilai data fitur minimum X_{min} dan dibagi dengan nilai data fitur maksimum X_{max} yang dikurang dengan X_{min}. Normalisasi data fitur dihitung dengan menggunakan Persamaan 3.8.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3.8}$$

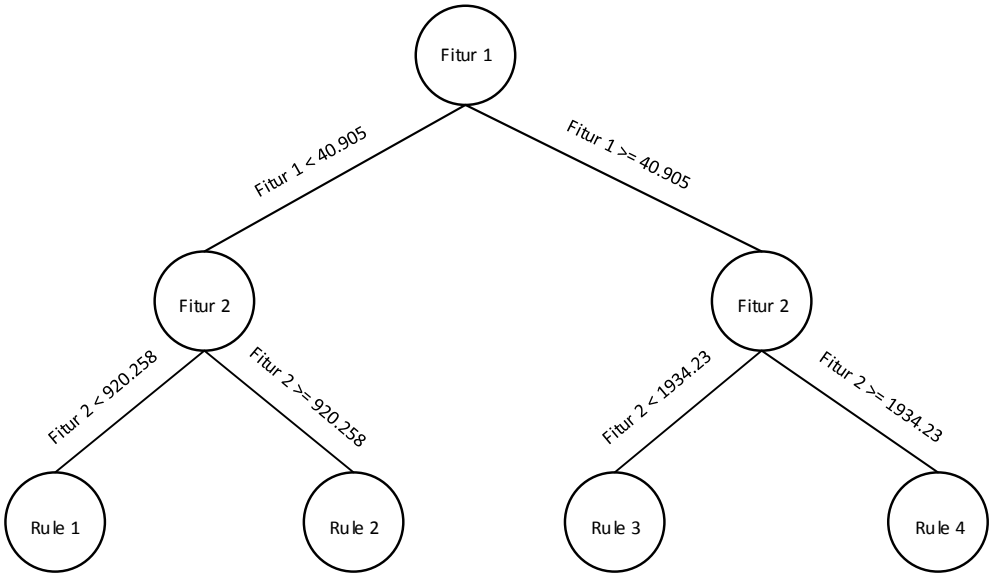
3.5 Kalibrasi

Kalibrasi adalah proses yang dilakukan karena proses pengambilan data latih dilakukan dengan menggunakan tangan penulis, sedangkan proses uji coba akan dilakukan oleh variasi pengguna, sehingga ada perbedaan ukuran antara tangan penulis dengan tangan pengguna. Proses kalibrasi dilakukan dengan cara membandingkan data fitur yang didapat dari tangan pengguna dengan data fitur yang didapat dari tangan penulis menggunakan Persamaan 3.9. Setelah mendapatkan *multiplierM* (nilai pengali), maka setiap fitur pengguna beracuan pada sumbu tertentu yang dikalikan dengan nilai pengali. *M* didapatkan dari nilai lebar atau nilai panjang tangan pengguna $N_{pengguna}$ dibagi dengan nilai lebar atau nilai panjang tangan penulis $N_{penulis}$.

$$M = \frac{N_{pengguna}}{N_{penulis}} \tag{3.9}$$

3.6 *Rule Based*

Sebelum proses klasifikasi data fitur dilakukan pengelompokan data fitur kedalam empat kelompok karena ketika L-GCNN digunakan pada data dengan jumlah kelas yang banyak akan terjadi *overfitting* yaitu sulit untuk menentukan kelas pada data. Proses pengelompokan ini dilakukan dengan tujuan untuk mengurangi terjadinya *overvitting* dan meningkatkan akurasi pada klasifikasi L-GCNN.



Gambar 3.7 Pembagian *rule* berdasarkan *Decission tree.*

Pengelompokan data fitur pada penelitian ini dilakukan dengan menggunakan metode *Decission tree*. Data fitur yang akan diklasifikasi akan dikelompokkan kedalam empat kelompok berdasarkan rule yang didapatkan dari *node* pada *decision tree.* Pembentukan *rule based* dilakukan dengan menggunakan *toolbox* pada Matlab 2015.

Tabel 3.2*Pseudocode* pembagian rule

```
if fitur 1 <40.905 && fitur 2 < 920.285
    Fitur termasuk ke rule 1
else if fitur 1 < 40.905 && fitur 2 >= 920.285
    Fitur termasuk ke rule 2
else if fitur 1 >= 40.905 && fitur 2 < 1934.23
    Fitur termasuk ke rule 3
else if fitur 1 >= 40.905 && fitur 2 >= 1934.23
    Fitur termasuk ke rule 4
```

Seperti ditunjukkan pada Gambar 3.7terdapat tujuh *node* dalam bentuk *decision tree,* dimana bentuk *decision tree* tersebut menggambarkan pembentukan *rule* kedalam empat *rule*. *Rule* pertama adalah dataset yang memenuhi syarat *rule* 1, *rule* kedua adalah dataset yang memenuhi syarat *rule* 2, *rule* ketiga adalah

dataset yang memenuhi syarat *rule*3, dan *rule* keempat adalah dataset yang memenuhi syarat *rule* 4. Hasil pembagian*rule* dengan menggunakan *decision tree* pada penelitian ini dapat dilihat padaTabel 3.2.

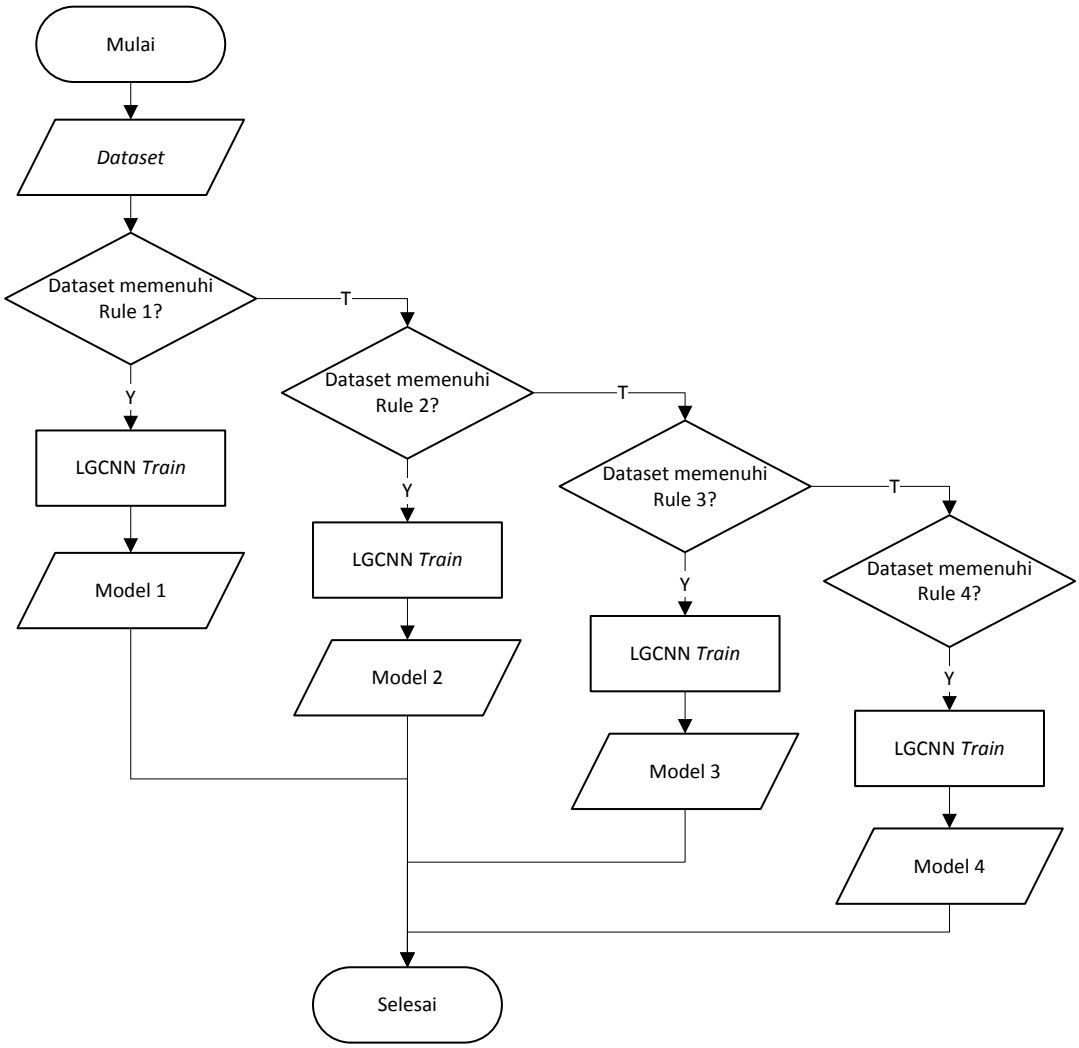
3.7 Klasifikasi

Pada tahapan klasifikasi ini bertujuan untuk mengklasifikasi data uji coba yang diinput secara *real time* dari LMCterhadap data latih yang sudah diberikan kelas atau label. Setiap data uji coba yang diinput akan diklasifikasikan terhadap data latih yang ada untuk mendapatkan kelas atau label pada data uji coba. Untuk proses klasifikasi akan digunakan metode klasifikasi L-GCNN karena *metode* tersebut memiliki keandalan dalam proses klasifikasi dan memiliki akurasi yang tinggi yang lebih baik dibandingkan dengan KNN dan SVM. Ada dua proses klasifikasi yaitu proses pelatihan data dan proses uji coba.

3.7.1 Proses Pelatihan

Proses pelatihan merupakan proses untuk pelatihan dataset dengan menggunakan metode L-GCNN. Sebelum dilakukan pelatihan terhadap dataset, dataset dibagi kedalam empat *rule* berdasarkan *node* pada *decision tree*. Adapun diagram proses pelatihan dataset dapat dilihat padaGambar 3.8. Untuk pelatihan dataset dengan menggunakan L-GCNN dapat dilihat pada Algoritma 2.1.

Pada proses pelatihan dataset, dataset akan dibagi kedalam empat rule. Dataset yang memenuhi syarat *rule* 1 maka dataset dilatih dengan menggunakan L-GCNN dan hasil *output* dari L-GCNN akan disimpan kedalam model 1. Dataset yang memenuhi syarat *rule* 2 maka dataset akan dilatih dengan menggunakan L-GCNN dan hasil *output* dari L-GCNN akan disimpan kedalam model 2.Dataset yang memenuhi syarat *rule* 3 maka dataset akan dilatih dengan menggunakan L-GCNN dan hasil *output* dari L-GCNN akan disimpan kedalam model 3. Dataset yang memenuhi syarat *rule* 4 maka dataset akan dilatih dengan menggunakan L-GCNN dan hasil *output* dari L-GCNN akan disimpan kedalammodel 4. File model 1, model 2, model 3 dan model 4 akan digunakan pada proses uji coba dengan data uji coba berdasarkan *rule*-nya.



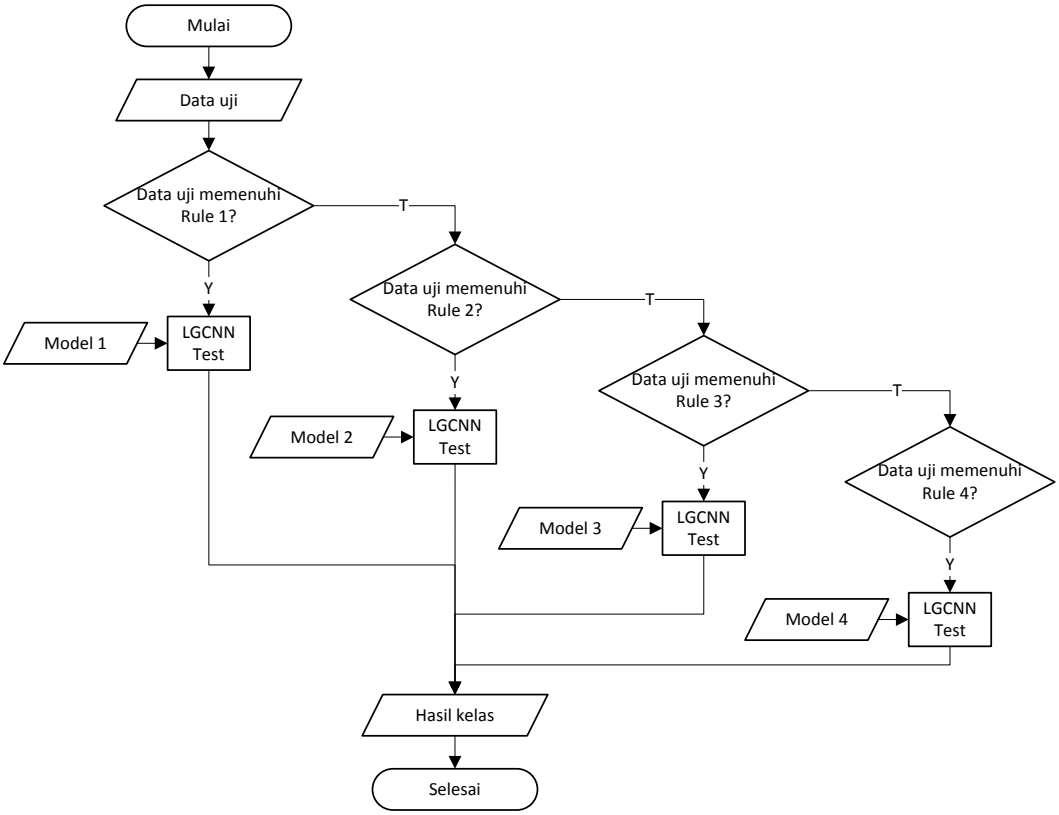
Gambar 3.8 Proses pelatihan dataset.

3.7.2 Proses Uji Coba

Proses uji coba merupakan langkah untuk mendapatkan kelas pada data uji coba. Proses uji coba dilakukan dengan menggunakan metode RB-L-GCNN. Untuk data uji coba yang akan diuji coba akan dicek apakah data tersebut memenuhi syarat *rule*1, *rule*2, *rule*3atau *rule*4. Untuk diagram proses uji coba dapat dilihat pada Gambar 3.9.

Jika data uji coba memenuhi syarat *rule*1 maka data uji coba akan di uji coba dengan menggunakan L-GCNN dan menggunakan dataset model 1. Jika data uji coba memenuhi syarat *rule*2 maka data uji coba akan di uji coba dengan

menggunakan L-GCNN dan menggunakan dataset model 2. Jika data uji coba memenuhi syarat *rule*3 maka data uji coba akan di uji coba dengan menggunakan L-GCNN dan menggunakan dataset model 3. Jika data uji coba memenuhi syarat *rule*4 maka data uji coba akan di uji coba dengan menggunakan L-GCNN dan menggunakan dataset model 4. Untuk proses uji coba dengan menggunakan L-GCNN dapat dilihat pada Algoritma 2.2.



Gambar 3.9 Proses uji coba.

[*Halaman ini sengaja dikosongkan*]

BAB 4
HASIL UJI COBA DAN PEMBAHASAN

Bab ini membahas pengujian dan analisa pada metode penelitian yang diusulkan dan perbandingan dengan beberapa metode lainnya. Pengujian yang dilakukan adalah pengujian terhadap kebutuhan fungsional secara keseluruhan. Pengujian dilakukan dengan beberapa skenario yaitu pengujian secara manual dan pengujian secara *real time*. Hasil analisa menjelaskan tentang rangkuman analisa dari hasil pengujian pada metode yang diusulkan pada akhir bab ini.

4.1 Lingkungan Uji Coba

Data Uji CobaSebagai uji coba pada penelitian ini, data sampel diujikan dengan menggunakan komputer (laptop) Compaq Presario CQ43 dengan dukungan processor Intel(R) Pentium(R) CPU B940 @ 2.00GHz (2 CPUs), ~2GHz, kapasitas RAM 4096MB, kapasitas *hardisk* 500 GB dan sebuah perangkat keras *LMC*. Perangkat lunak pendukung adalah sistem operasi *Windows* 10 *Pro*,*NetBeans* IDE 8.0.2 dan SDK *LeapDeveloperKit*_2.3.1+31549_*win*.

Perekaman dataset dan uji coba *real time* dilakukan pada tempat dengan pencahayaan yang stabil, karena pencahayaan yang tidak stabil akan mempengaruhi sensor pada LMC dalam proses perekaman titik-titik koordinat pada tangan. *Leap motion visualizer* adalah API dari *developer* yang men-visualisasikan bentuk tangan dalam animasi 3D yang dapat digunakan sebagai acuan dalam perekaman titik-titik koordinat tangan. Ketika visualisasi tangan dalam animasi sama bentuknya dengan tangan yang dideteksi maka titik-titik koordinat tangan yang direkam benar, karena titik-titik koordinat tangan yang direkam berdasarkan pada visualisasi tangan pada *Leap Motion Visualizer*.

4.2 Data Uji Coba

Dataset pengenalan bahasa isyarat SIBI ini dilakukan dengan menggunakan data sampel yang direkam dengan menggunakan tangan kanan penulis sebanyak 10 sampel untuk setiap huruf. Jumlah sampel dengan 26 huruf (A – Z) dan satu posisi tangan normal, sehingga jumlah total sampel sebanyak 270 sampel. Untuk

uji coba manual, data uji coba yang digunakan adalah data sampel itu sendiri, sedangkan data uji coba yang digunakan untuk uji coba *real time* adalah data uji coba yang direkam langsung dari tangan penulis.

4.3 Skenario Uji Coba dan Evaluasi

Hasil Skenario Uji CobSecara garis besar skenario uji coba dilakukan dengan dua skenario uji coba yang dapat penulis definisikan sebagai skenario A dan skenario B. Skenario A adalah skenario uji coba yang dilakukan dengan cara manual, sedangkan skenario B adalah skenario uji coba yang dilakukan dengan cara *real time.*

Skenario A dibagi kedalam dua sub skenario yaitu skenario A1 dan skenario A2 sebagai berikut :

1. Skenario A1 adalah skenario ujicoba manual yang dilakukan dengan hanya menggunakan fitur statis yang didapatkan dari LMC.
2. Skenario A2 adalah skenario uji coba yang dilakukan dengan menggunakan kombinasi fitur statis dan fitur dinamis yang didapatkan dari LMC.

Skenario A1 dan skenario A2 dilakukan dengan menggunakan beberapa metode klasifikasi untuk pengenalan bahasa isyarat SIBI yaitu metode yang diusulkan RB-L-GCNN yang akan dibandingkan dengan metode sebelumnya L-GCNN dan beberapa metode lainnya *Decission Tree*, SVM dan KNN. Pada uji coba dengan menggunakan SVM, kernel yang akan digunakan adalah kernel linear. Sedangkan pada uji coba dengan menggunakan KNN, nilai K yang digunakan adalah 1.

Skenario B dibagi kedalam dua sub skenario yaitu skenario B1 dan skenario B2 sebagai berikut :

1. Skenario B1 adalah skenario uji coba *real time* yang dilakukan dengan hanya menggunakan fitur statis yang didapatkan dari LMC.
2. Skenario B2 adalah skenario uji coba *real time* yang dilakukan dengan menggunakan kombinasi fitur statis dan fitur dinamis yang didapatkan dari LMC.

Skenario uji coba B1 dan B2 dilakukan dengan menggunakan metode klasifikasi untuk pengenalan bahasa isyarat SIBI yaitu metode yang diusulkan RB-L-GCNN yang akan dibandingkan dengan metode sebelumnya L-GCNN.

Untuk evaluasi hasil uji coba yang dilakukan akan diukur dengan menggunakan akurasi pengenalan. Akurasi dihitung dengan membandingkan data pengenalan bahasa isyarat yang benar dengan jumlah total data yang diuji. Untuk menghitung akurasi pengenalan digunakan Persamaan 4.1.

$$Akurasi = \frac{N_{cor}}{N_{tes}}\ x\ 100\%, \qquad (4.1)$$

dimana N_{cor} adalah jumlah data ujicoba yang terklasifikasi dengan benar danN_{tes} adalah total jumlah data yang diuji coba.

4.3.1 Hasil Skenario Uji Coba A dan Analisa

Skenario uji coba A dilakukan dalam dua skenario uji coba dimana skenario uji coba A1 dilakukan dengan menggunakan fitur statis, sedangkan sekenario uji coba A2 dilakukan dengan menggunakan kombinasi fitur statis dengan fitur dinamis. Kedua skenario uji coba ini dilakukan dengan tujuan untuk mengetahui perbandingan akurasi antara pengenalan bahasa isyarat SIBI dengan hanya menggunakan fitur statis dengan metode usulan yaitu pengenalan bahasa isyarat sibi dengan menggunakan kombinasi fitur statis dengan fitur dinamis.

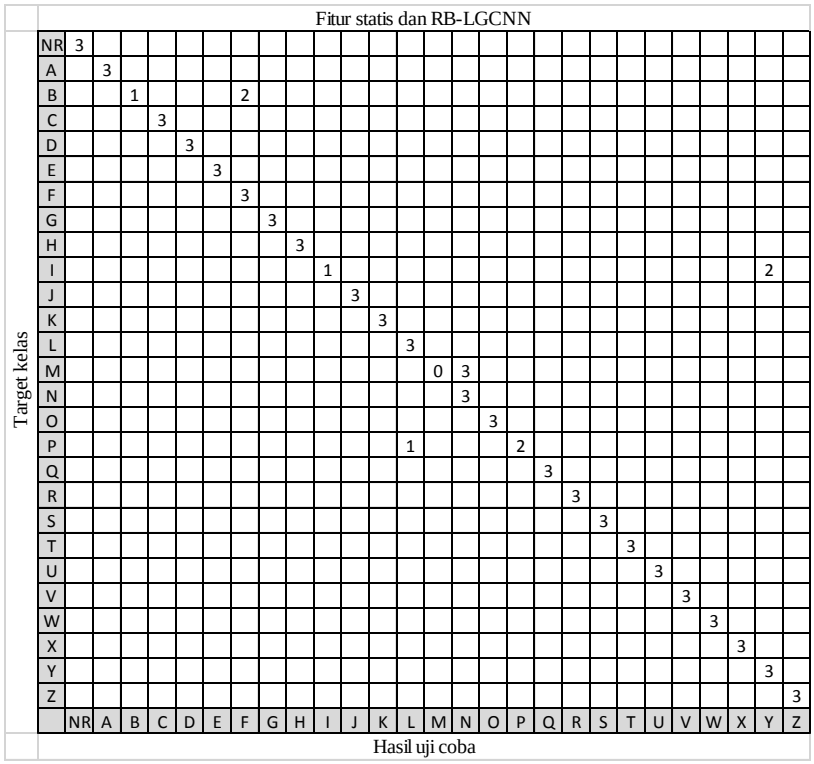
Pada kedua skenario ini juga dilakukan uji coba untuk mengetahui perbandingan akurasi pengenalan bahasa isyarat SIBI dengan menggunakan metode L-GCNN dengan metode usulan yaitu metode RB-L-GCNN, serta perbandingan dengan beberapa metode lainnya yaitu *Decission tree*, KNN dan SVM. Skenario uji coba A dilakukan dengan membagi dataset sebagai data training sebanyak 70% dan data testing sebanyak 30%. Dataset yang digunakan adalah dataset yang direkam dengan menggunakan tangan penulis.

4.3.1.1 Hasil Skenario Uji Coba A1 dan Analisa

Skenario uji coba A1 dilakukan secara manual dengan menggunakan fitur statis dari *LMC* untuk mengetahui akurasi pengenalan bahasa isyarat SIBI.

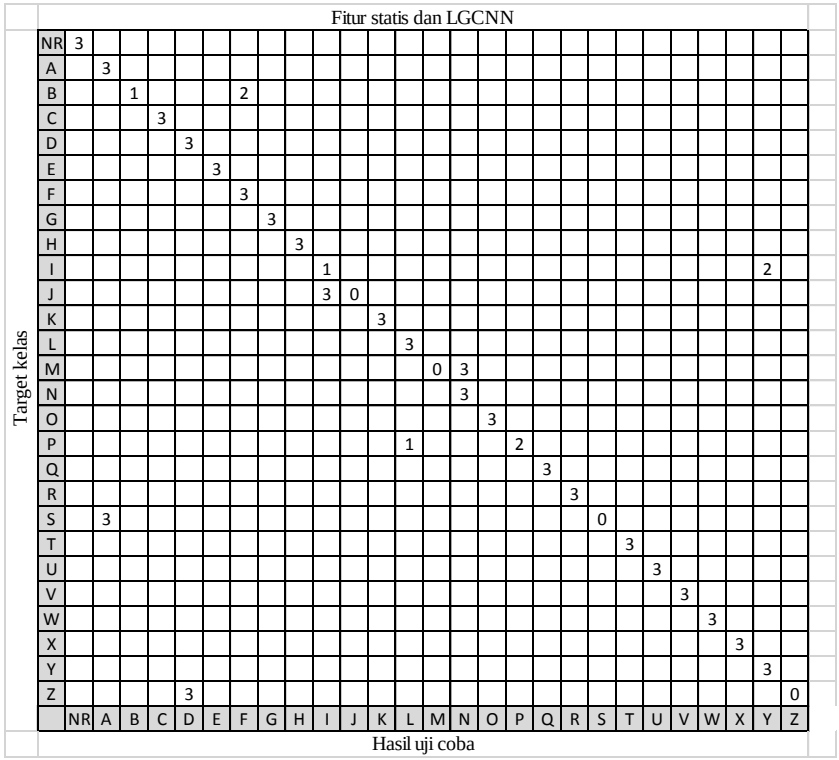
Sedangkan metode klasifikasi yang digunakan adalah L-GCNN, RB-L-GCNN, *Decission Tree*, SVM dan KNN. Skenario uji coba A1 dilakukan dengan membagi dataset sebagai data training sebanyak 70% dan sebagai data testing sebanyak 30%.

Untuk uji coba skenario A1 dengan menggunakan fitur statis dan metode L-GCNN dan RB-L-GCNN dilakukan dengan menggunakan aplikasi yang dibuat dengan menggunakan bahasa pemrograman java. Untuk hasil uji coba skenario A1 menggunakan metode RB-L-GCNN dapat dilihat pada Gambar 4.1dan L-GCNN dapat dilihat padaGambar 4.2.



Gambar 4.1 Hasil uji coba fitur statis dan RB-L-GCNN

Pada Gambar 4.1ditunjukkan bahwa hasil uji coba dengan menggunakan fitur statis dan metode RB-L-GCNN memiliki akurasi rata-rata 89.88%. Ada beberapa huruf yang tidak dapat dikenal dengan baik seperti huruf B, I, M, dan P, Hal ini terjadi karena huruf-huruf tersebut memiliki bentuk isyarat yang serupa sehingga fitur yang dihasilkan memiliki tingkat kemiripan yang tinggi.



Gambar 4.2 Hasil uji coba fitur statis dan L-GCNN

Dari Gambar 4.2 ditunjukkan bahwa hasil uji coba dengan menggunakan fitur statis dan metode L-GCNN memiliki akurasi rata-rata 86.42%. Ada beberapa huruf yang tidak dapat dikenal dengan baik seperti huruf B, I, dan P. Hal ini terjadi karena huruf-huruf tersebut memiliki fitur yang hampir mirip. Sedangkan beberapa huruf seperti huruf J, M, S, dan Z tidak dapat dikenal sama sekali, hal ini terjadi karena huruf-huruf tersebut memiliki bentuk isyarat yang serupa sehingga fitur yang dihasilkan memiliki tingkat kemiripan yang tinggi.

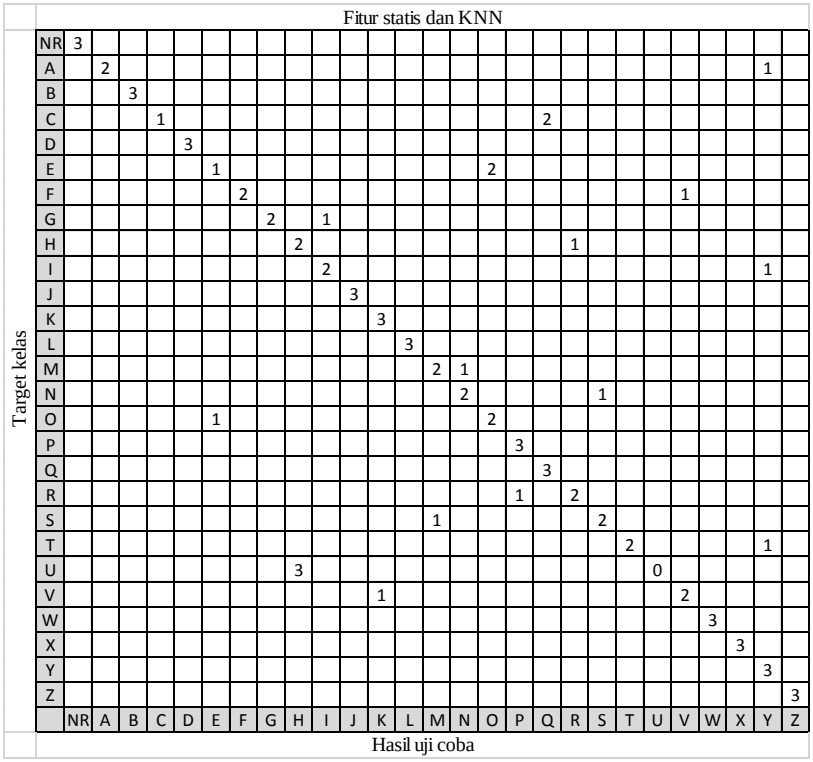
Untuk uji coba skenario A1 tahap pertama dengan menggunakan fitur statis dan metode *Decission tree,* SVM dan KNN di uji coba dengan menggunakan *Toolbox* matlab 2015. Hasil uji coba dengan menggunakan *Decision tree* dapat dilihat pada Gambar 4.3, Hasil uji coba dengan menggunakan SVM pada Gambar 4.4 dan Hasil uji coba dengan menggunakan KNN pada Gambar 4.5.

Fitur statis dan Decision tree																												
Target kelas	NR	3																										
	A		2																								1	
	B			3																								
	C			2	0													1										
	D					3																						
	E		1				1									1												
	F							2					1															
	G								2									1										
	H									3																		
	I				1						0							2										
	J											3																
	K												3															
	L													3														
	M														3													
	N															3												
	O						1										2											
	P																	3										
	Q																		3									
	R									2										1								
	S														1	2					0							
	T																		2			0					1	
	U									2										1			0					
	V																							3				
	W																								3			
	X																									3		
	Y																		1								2	
	Z																											3
		NR	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
	Hasil uji coba																											

Gambar 4.3 Hasil uji coba fitur statis dan *Decission tree*

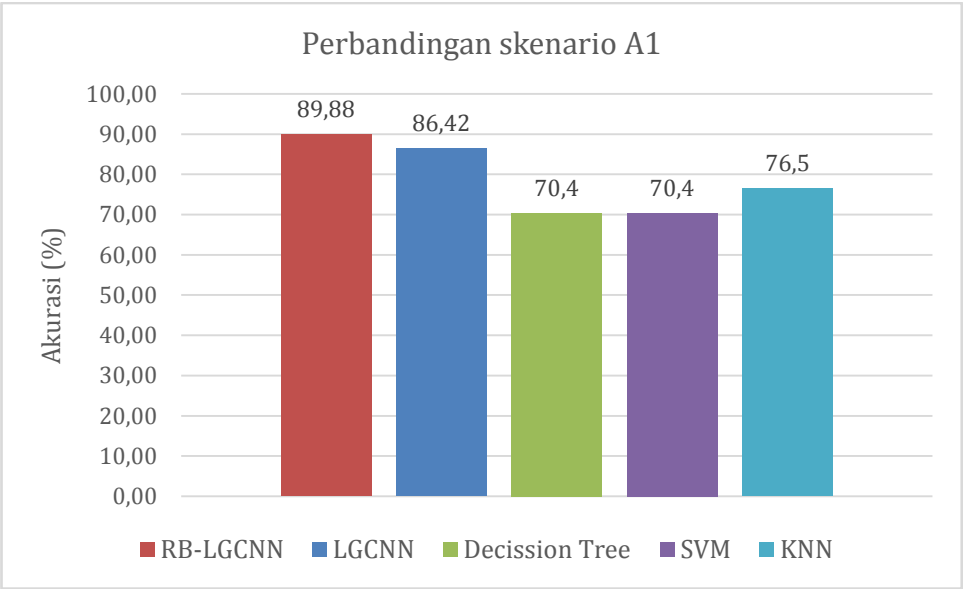
Fitur statis dan SVM																												
Target kelas	NR	3																										
	A		2				1																					
	B			3																								
	C				1						2																	
	D					3																						
	E						1										2											
	F							2																1				
	G								2		1																	
	H									3																		
	I										1							1			1							
	J											2										1						
	K												3															
	L													3														
	M														1						2							
	N															3												
	O						1										2											
	P																	3										
	Q																		3									
	R									1										2								
	S														1	2					0							
	T																					2					1	
	U									2										1			0					
	V												1											2					
	W																								3			
	X																									3		
	Y																					1					2	
	Z																						1					2
		NR	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
	Hasil uji coba																											

Gambar 4.4 Hasil uji coba fitur statis dan SVM dengan kernel linier.



Gambar 4.5 Hasil uji coba fitur statis dan KNN dengan K=1.

Dari beberapa percobaan pada skenario uji coba A1, hasil akurasi pengenalan dapat dibandingkan seperti pada Gambar 4.6.



Gambar 4.6Hasil perbandingan skenario A1

Dari Gambar 4.6 dapat dilihat bahwa pengenalan huruf dengan menggunakan fitur statis dan RB-L-GCNN memiliki akurasi pengenalan lebih baik dibandingkan dengan L-GCNN, *decision tree*, SVM dan KNN. *Rule based* dapat meningkatkan akurasi pengenalan huruf pada metode L-GCNN hingga 4.07%.

4.3.1.2 Hasil Skenario Uji Coba A2 dan Analisa

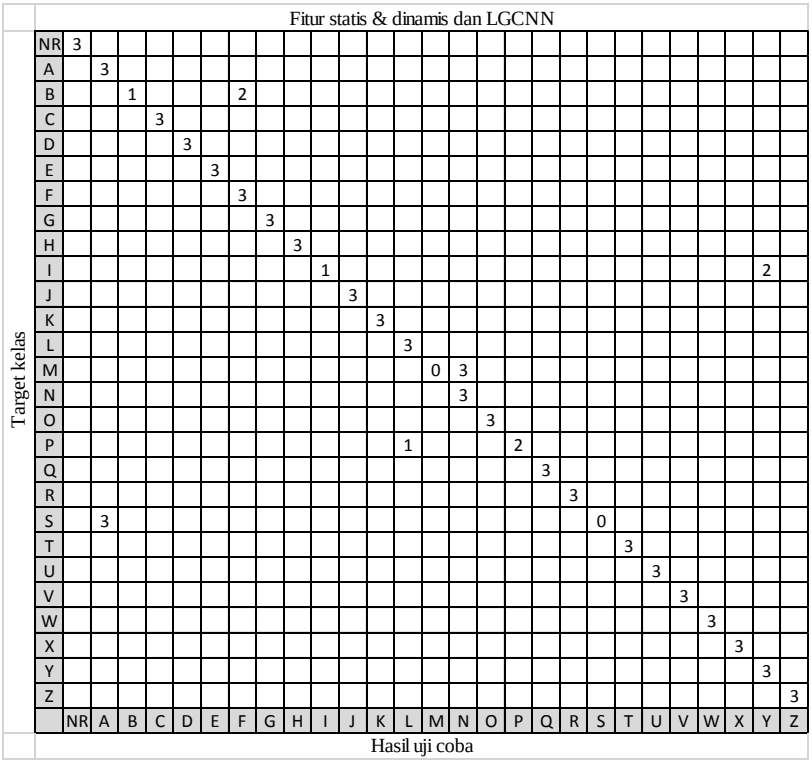
Uji coba dengan cara manual dilakukan dengan membandingan metode L-GCNN, RB-L-GCNN dan Decision Tree. Skenario uji coba A2 dilakukan dengan membagi dataset sebagai data training sebanyak 70% dan sebagai data testing sebanyak 30%.

Untuk skenario uji coba A2 tahap pertama dengan menggunakan kombinasi fitur statis dan dinamis dengan metode L-GCNN dan RB-L-GCNN dilakukan dengan menggunakan aplikasi yang dibuat dengan menggunakan java. Hasil skenario uji coba A2 dengan menggunakan metode RB-L-GCNN dapat dilihat pada Gambar 4.7dan L-GCNN dapat dilihat padaGambar 4.8.

Fitur Statis & Dinamis dan RB-LGCNN

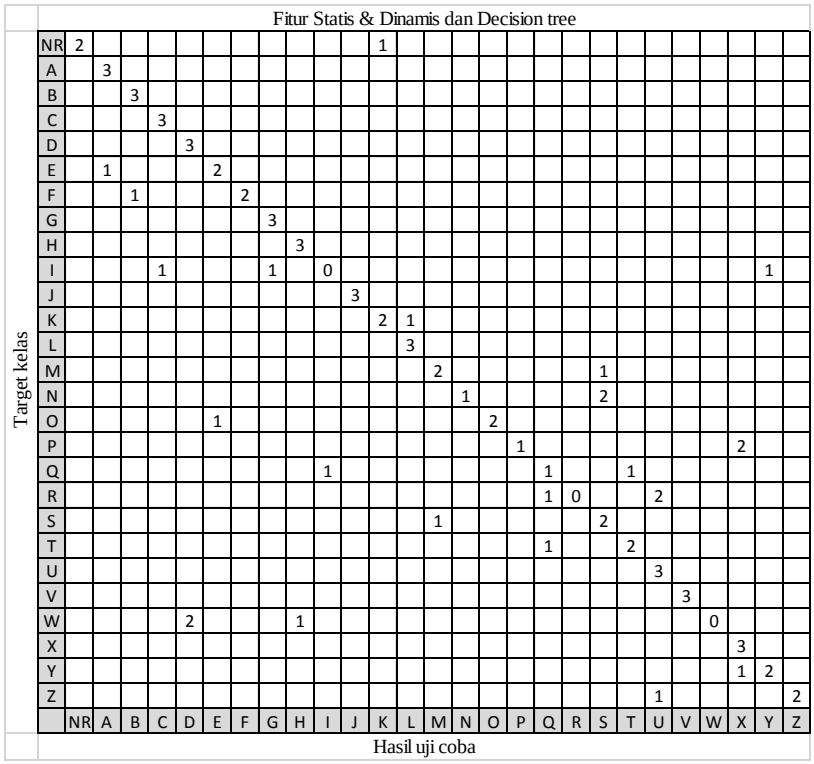
Target kelas \ Hasil uji coba	NR	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
NR	3																										
A		3																									
B			1				2																				
C				3																							
D					3																						
E						3																					
F							3																				
G						3		0																			
H									3																		
I						2				1																	
J						3					0																
K												3															
L													3														
M														0	3												
N															3												
O																3											
P													1				2										
Q																		3									
R																			3								
S																				3							
T						2															1						
U																						3					
V																							3				
W																								3			
X																									3		
Y						2																				1	
Z	1																										2

Gambar 4.7 Hasil uji coba kombinasi fitur statis dengan dinamis dan RB-L-GCNN



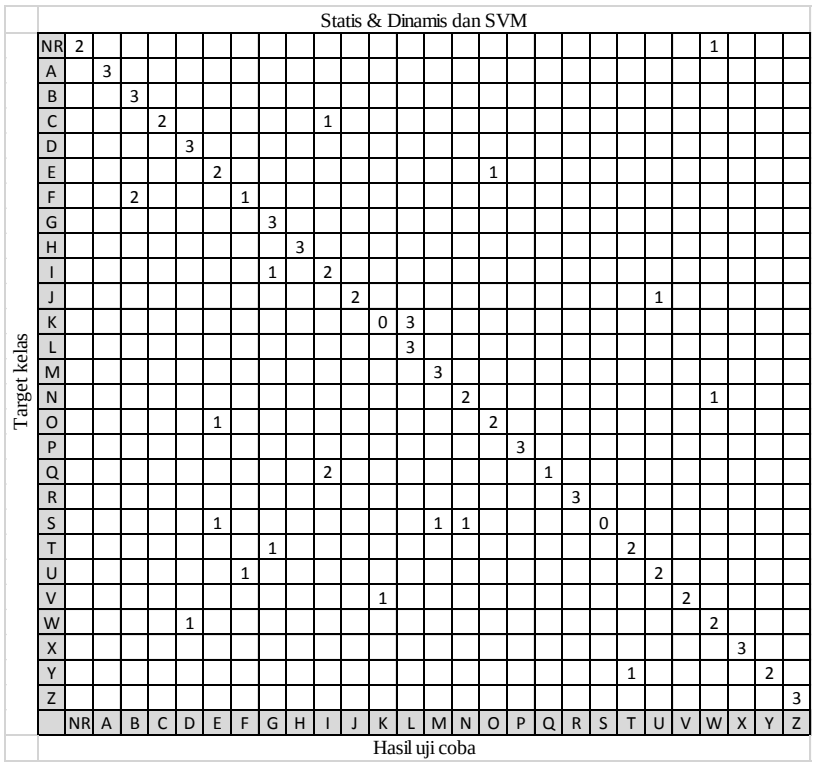
Gambar 4.8 Hasil uji coba kombinasi fitur statis dengan dinamis dan L-GCNN

Untuk skenario uji coba A2 dengan menggunakan kombinasi fitur statis dan fitur dinamis dengan metode *Decission tree*, SVM dan KNN dilakukan dengan menggunakan *toolbox* matlab 2015. Untuk uji coba dengan menggunakan metode SVM akan digunakan kernel standard yaitu kernel linear, sedangkan untuk uji coba dengan menggunakan metode KNN maka akan digunakan dengan nilai K adalah 1. Hasil skenario uji coba A2 dengan menggunakan metode *Decision tree* dapat dilihat padaGambar 4.9, hasil scenario uji coba A2 dengan menggunakan metode SVM dapat dilihat pada Gambar 4.10dan hasil scenario uji coba dengan menggunakan KNN dapat dilihat padaGambar 4.11.



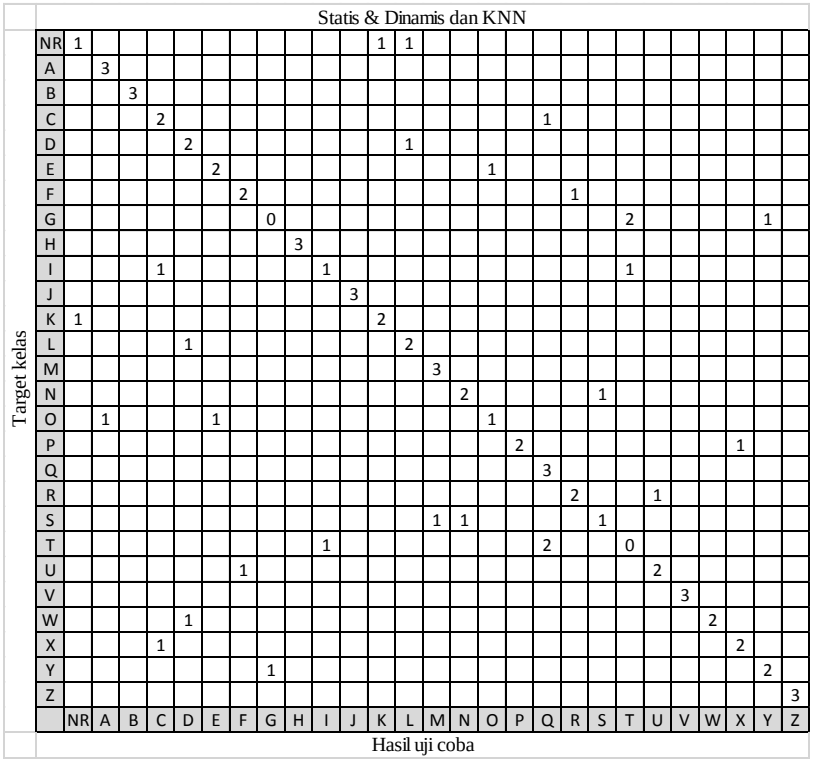
Target kelas \ Hasil uji coba	NR	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
NR	2											1															
A		3																									
B			3																								
C				3																							
D					3																						
E		1				2																					
F			1				2																				
G								3																			
H									3																		
I				1				1		0																1	
J											3																
K												2	1														
L													3														
M														2						1							
N															1					2							
O						1										2											
P																	1								2		
Q										1								1			1						
R																		1	0		2						
S														1						2							
T																		1			2						
U																						3					
V																							3				
W					2				1															0			
X																									3		
Y																									1	2	
Z																						1					2

Gambar 4.9 Hasil uji coba kombinasi fitur statis dengan dinamis dan *Decission tree*.



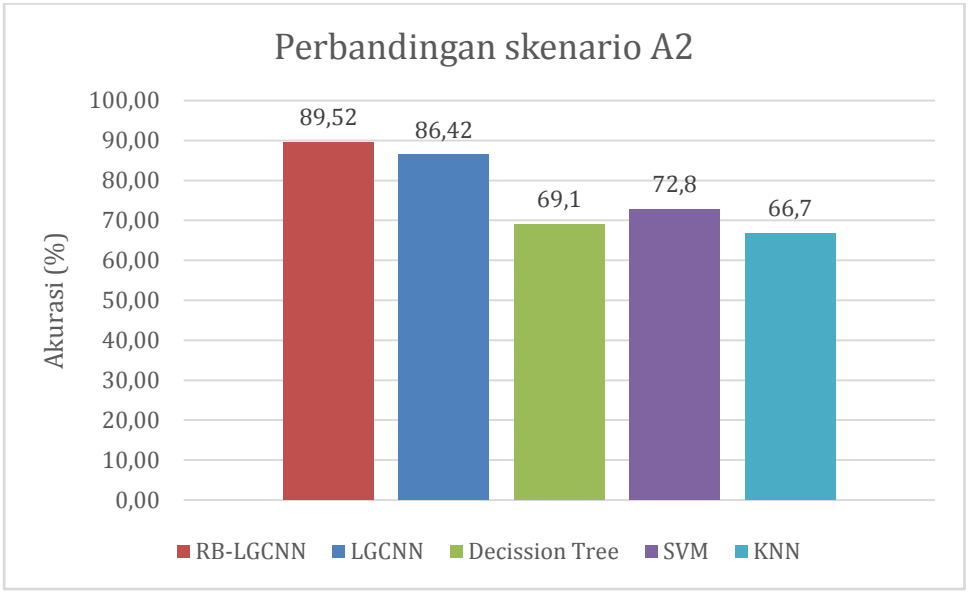
Target kelas \ Hasil uji coba	NR	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
NR	2																							1			
A		3																									
B			3																								
C				2						1																	
D					3																						
E						2										1											
F			2				1																				
G								3																			
H									3																		
I								1		2																	
J											2											1					
K												0	3														
L													3														
M														3													
N															2									1			
O						1										2											
P																	3										
Q										2								1									
R																			3								
S						1								1	1					0							
T								1													2						
U							1															2					
V												1											2				
W					1																			2			
X																									3		
Y																					1					2	
Z																											3

Gambar 4.10 Hasil uji coba kombinasi fitur statis dengan dinamis dan SVM dengan kernel linier.



Gambar 4.11 Hasil uji coba kombinasi fitur statis dengan dinamis dan KNN dengan K=1.

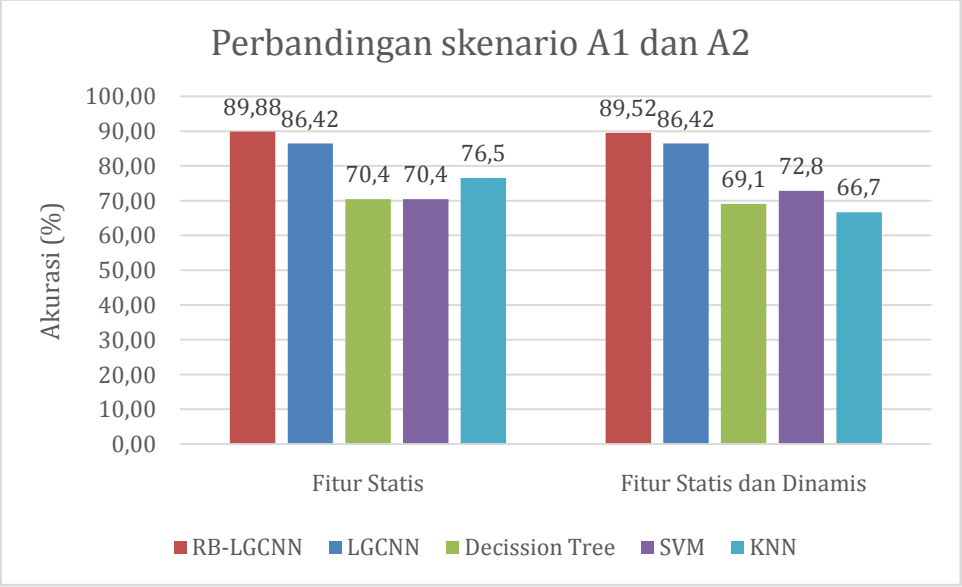
Dari beberapa percobaan pada skenario uji coba A2, hasil akurasi dapat dibandingkan seperti padaGambar 4.12.



Gambar 4.12Hasil perbandingan skenario A2

Dari Gambar 4.12dapat dilihat bahwa pengenalan huruf dengan menggunakan kombinasi fitur statis dengan fitur dinamis dan RB-L-GCNN memiliki akurasi pengenalan lebih baik dibandingkan dengan L-GCNN, *decision tree*, SVM dan KNN. *Rule based* dapat meningkatkan akurasi pengenalan huruf pada metode L-GCNN hingga 6.87%.

Dari hasil skenario uji coba A1 dan A2 dapat dibandingkan antara penggunaan fitur statis dengan kombinasi fitur statis dengan fitur dinamis seperti pada Gambar 4.13.



Gambar 4.13Hasil perbandingan skenario A1 dan A2.

Dari Gambar 4.13dapat dilihat bahwa akurasi pengenalan huruf dengan menggunakan metode RB-L-GCNN memiliki akurasi yang lebih baik dibandingkan dengan metode L-GCNN, *Decision tree*, SVM dan KNN.

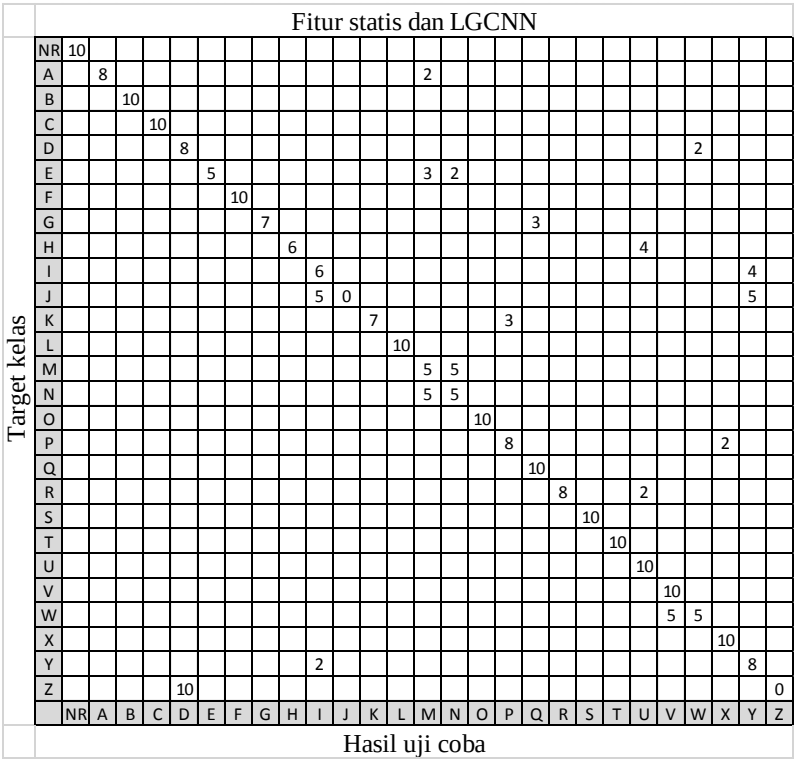
4.3.2 Hasil Skenario Uji Coba B dan Analisa

Skenario uji coba B adalah skenario uji coba yang dilakukan secara *real time.* Dalam skenario B dibagi menjadi dua sub skenario yaitu skenario B1 dan skenario B2. Skenario B1 adalah uji coba yang dilakukan dengan menggunakan fitur statis dan perbandingan dari metode L-GCNN dan RB-L-GCNN. Sedangkan

skenario B2 adalah uji coba yang dilakukan dengan menggunakan kombinasi fitur statis dan fitur dinamis dan perbandingan dari metode L-GCNN dan RB-L-GCNN.

4.3.2.1 Hasil Skenario Uji Coba B1 dan Analisa

Untuk skenario uji coba B1 tahap pertama, pengujian dilakukan secara *real time* dengan menggunakan fitur statis dan metode L-GCNN. Pengujian untuk setiap huruf dilakukan sebanyak sepuluh kali dengan menggunakan tangan penulis. Hasil uji coba skenario B1 tahap pertama dapat dilihat pada Gambar 4.14.

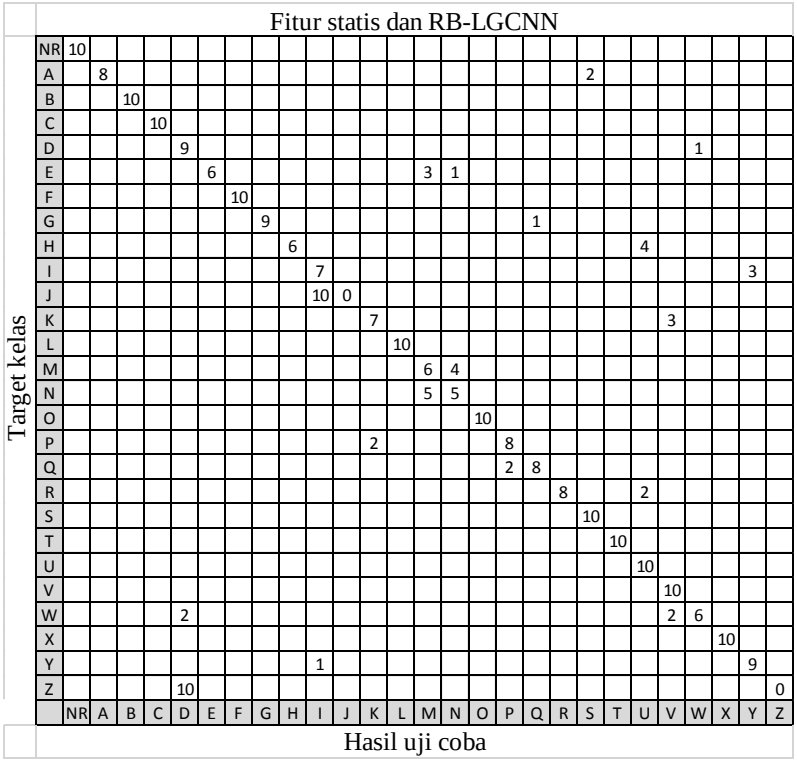


Gambar 4.14 Hasil skenario uji coba B1 tahap pertama

Pada Gambar 4.14 dapat dilihat bahwa akurasi pengenalan cukup baik untuk beberapa huruf seperti huruf NR, B, C, F, G, L, O, Q, S, T, U, V dan X karena huruf tersebut memiliki fitur yang tingkat kemiripannya cukup rendah dibandingkan dengan huruf-huruf yang lain. Beberapa kali terjadi kesalahan pengenalan pada huruf A, E, M dan N karena keempat huruf tersebut memiliki tingkat kemiripan fitur yang sangat tinggi. Ada dua huruf J dan Z yang bersifat

dinamis yang tidak dapat dikenal dengan baik karena kedua huruf tersebut bersifat dinamis, sedangkan fitur yang digunakan hanya fitur yang bersifat statis.

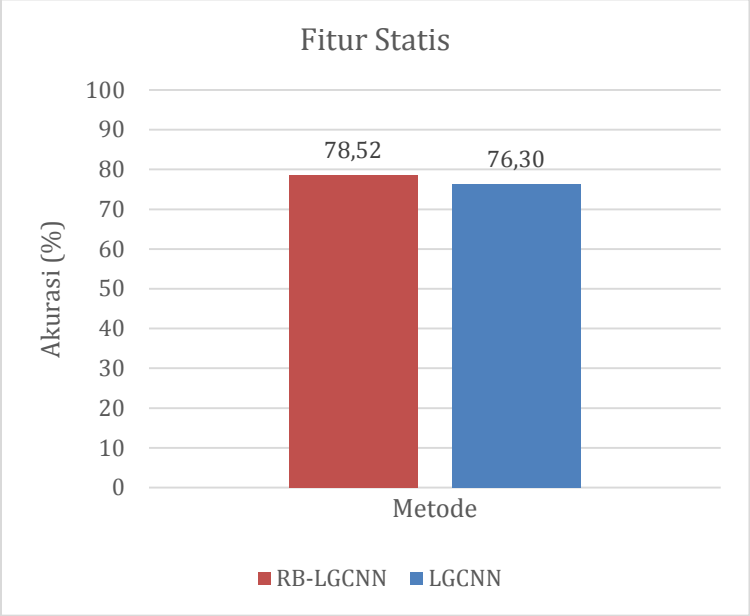
Untuk skenario uji coba B1 tahap kedua, pengujian dilakukan secara *real time* dengan menggunakan fitur statis dan metode RB-L-GCNN. Pengujian untuk setiap huruf dilakukan sebanyak sepuluh kali dengan menggunakan tangan penulis. Hasil uji coba skenario B1 tahap kedua dapat dilihat pada Gambar 4.15.



Gambar 4.15 Hasil skenario uji coba B1 tahap kedua

Pada Gambar 4.15 dapat dilihat bahwa akurasi pengenalan cukup baik untuk beberapa huruf seperti huruf NR, B, C, F, G, L, O, Q, S, T, U, V dan X karena huruf tersebut memiliki fitur yang tingkat kemiripannya cukup rendah dibandingkan dengan huruf-huruf yang lain. Beberapa kali terjadi kesalahan pengenalan pada huruf A, E, M dan N karena keempat huruf tersebut memiliki tingkat kemiripan fitur yang sangat tinggi. Ada dua huruf J dan Z yang bersifat dinamis yang tidak dapat dikenal karena kedua huruf tersebut bersifat dinamis, sedangkan fitur yang digunakan hanya fitur yang bersifat statis.

Dari hasil skenario uji coba B1 tahap pertama dan skenario uji coba B1 tahap kedua dapat dilihat bahwa perbandingan akurasi pengenalan bahasa isyarat SIBI dengan menggunakan fitur statis dan metode RB-L-GCNN memiliki akurasi yang lebih baik dibandingkan dengan menggunakan fitur statis dan metode L-GCNN.



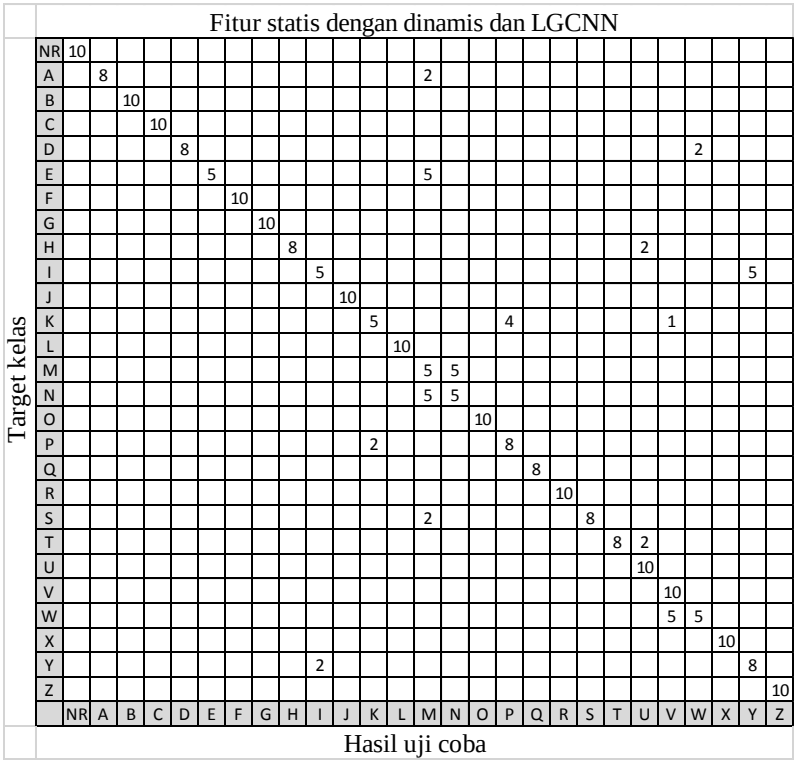
Gambar 4.16 Hasil perbandingan akurasi metode L-GCNN dan RB-L-GCNN dengan menggunakan fitur statis.

Dari Gambar 4.16 dapat dilihat bahwa akurasi pengenalan bahasa isyarat SIBI dengan menggunakan fitur statis dan RB-L-GCNN memiliki akurasi 78,52%, sedangkan pengenalan bahasa isyarat SIBI dengan menggunakan fitur statis dan L-GCNN memiliki akurasi 76,3%. RB-L-GCNN memiliki akurasi yang lebih baik dibandingkan dengan L-GCNN. Peningkatan akurasi ini terjadi karena pembentukan *rule based* pada L-GCNN dapat mengurangi terjadinya *overfitting* atau kesalahan dalam menentukan kelas pada data.

4.3.2.2 Hasil Skenario Uji Coba B2 dan Analisa

Untuk skenario uji coba B2 tahap pertama, pengujian dilakukan secara *real time* dengan menggunakan kombinasi fitur statis dengan fitur dinamis dan metode

L-GCNN. Pengujian untuk setiap huruf dilakukan sebanyak sepuluh kali dengan menggunakan tangan penulis. Hasil uji coba skenario B2 tahap pertama dapat dilihat pada Gambar 4.17.

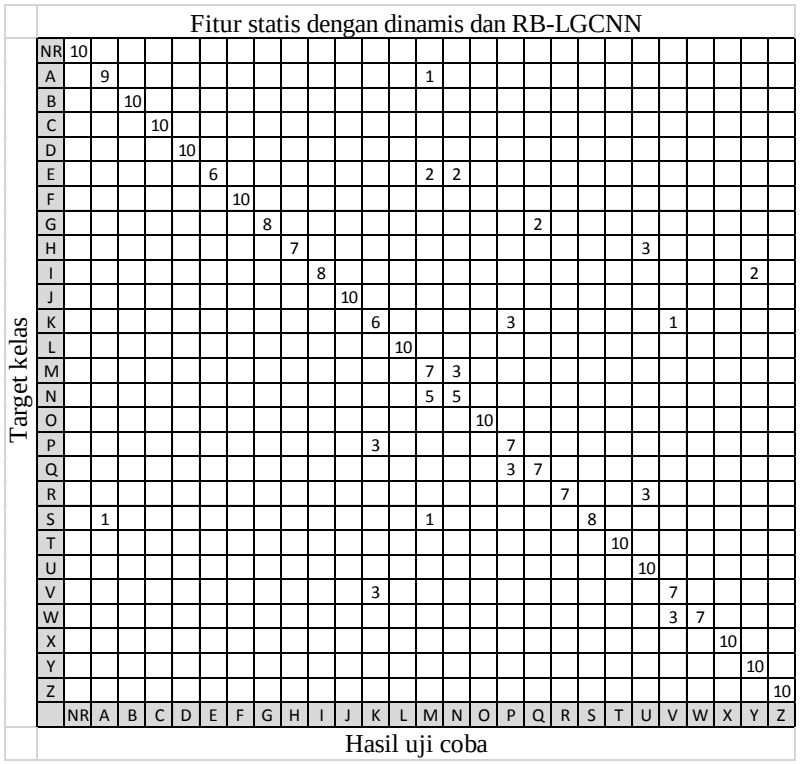


Gambar 4.17 Hasil skenario uji coba B2 tahap pertama.

Pada Gambar 4.17 dapat dilihat bahwa akurasi pengenalan cukup baik untuk beberapa huruf seperti huruf NR, B, C, F, G, L, O, Q, S, T, U, V dan X karena huruf tersebut memiliki fitur yang tingkat kemiripannya cukup rendah dibandingkan dengan huruf-huruf yang lain. Beberapa kali terjadi kesalahan pengenalan pada huruf A, E, M dan N karena keempat huruf tersebut memiliki tingkat kemiripan fitur yang sangat tinggi. Huruf yang bersifat dinamis dapat dikenal dengan baik seperti huruf J dan Z, karena fitur yang digunakan adalah kombinasi fitur statis dengan fitur dinamis.

Untuk skenario uji coba B2 tahap kedua, pengujian dilakukan secara *real time* dengan menggunakan kombinasi fitur statis dengan fitur dinamis dan metode RB-L-GCNN. Pengujian untuk setiap huruf dilakukan sebanyak sepuluh kali

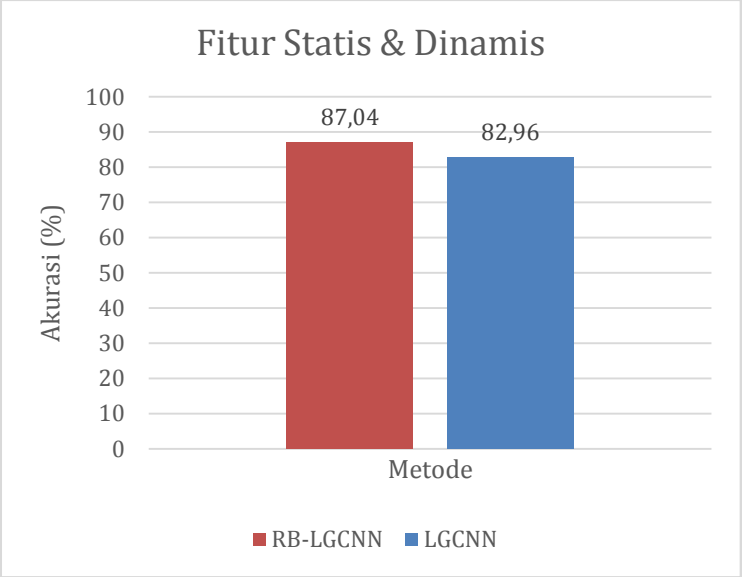
dengan menggunakan tangan penulis. Hasil uji coba skenario B2 tahap kedua dapat dilihat pada Gambar 4.18.



Gambar 4.18 Hasil skenario uji coba B2 tahap kedua

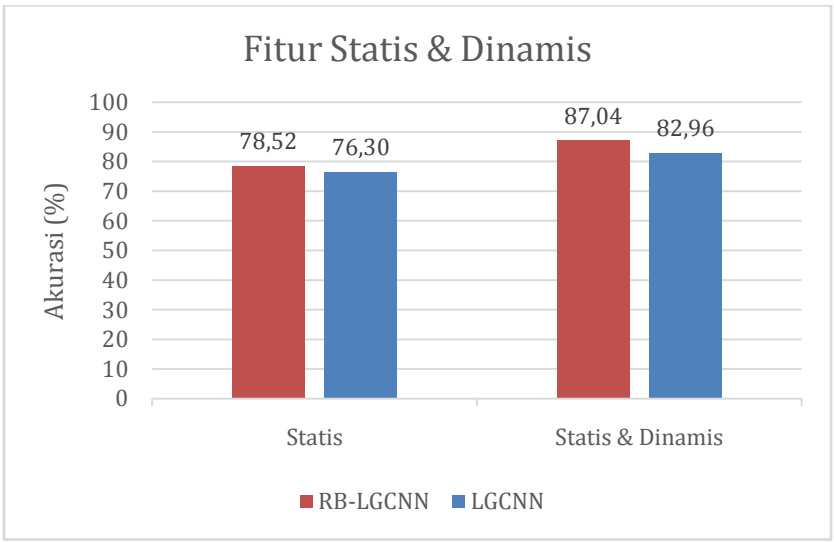
Pada Gambar 4.18 dapat dilihat bahwa akurasi pengenalan cukup baik untuk beberapa huruf seperti huruf NR, B, C, F, G, L, O, Q, S, T, U, V dan X karena huruf tersebut memiliki fitur yang tingkat kemiripannya cukup rendah dibandingkan dengan huruf-huruf yang lain. Beberapa kali terjadi kesalahan pengenalan pada huruf A, E, M dan N karena keempat huruf tersebut memiliki tingkat kemiripan fitur yang sangat tinggi. Huruf yang bersifat dinamis dapat dikenal dengan baik seperti huruf J dan Z, karena fitur yang digunakan adalah kombinasi fitur statis dengan fitur dinamis.

Dari hasil skenario uji coba B2 tahap pertama dan skenario uji coba B2 tahap kedua dapat dilihat bahwa perbandingan akurasi pengenalan bahasa isyarat SIBI dengan menggunakan kombinasi fitur statis dengan fitur dinamis dan metode RB-L-GCNN memiliki akurasi yang lebih baik dibandingkan dengan menggunakan kombinasi fitur statis dengan fitur dinamis dan metode L-GCNN.



Gambar 4.19 Hasil perbandingan akurasi metode L-GCNN dan RB-L-GCNN dengan menggunakan kombinasi fitur statis dan fitur dinamis.

Dari Gambar 4.19 dapat dilihat bahwa akurasi pengenalan bahasa isyarat SIBI dengan menggunakan fitur statis dan RB-L-GCNN memiliki akurasi 87,08%, sedangkan pengenalan bahasa isyarat SIBI dengan menggunakan fitur statis dan L-GCNN memiliki akurasi 82,96%. RB-L-GCNN memiliki akurasi yang lebih baik dibandingkan dengan L-GCNN. Peningkatan akurasi ini terjadi karena pembentukan *rule based* pada L-GCNN dapat mengurangi terjadinya *overfitting* atau kesalahan dalam menentukan kelas pada data.



Gambar 4.20 Hasil skenario B1 dan B2

Dari hasil skenario uji coba B1 dan B2 dapat disimpulkan pada grafik seperti pada Gambar 4.20. Pada Gambar 4.20 dapat dilihat bahwa ada dua perbandingan, dimana perbandingan pertama adalah pengenalan bahasa isyarat dengan menggunakan metode L-GCNN dan RB-L-GCNN. Sedangkan perbandingan yang kedua adalah pengenalan bahasa isyarat dengan menggunakan fitur statis dan menggunakan kombinasi fitur statis dan fitur dinamis.

Pengenalan bahasa isyarat dengan menggunakan fitur statis dan metode L-GCNN memiliki akurasi 76,30%. Sedangkan menggunakan fitur statis dan metode RB-L-GCNN memiliki akurasi 78,52%. Hal ini dapat dibandingkanbahwa pengenalan bahasa isyarat dengan menggunakan fitur statis dan metode RB-L-GCNN memiliki peningkatan akurasi 2,22% lebih baik dari L-GCNN.

Pengenalan bahasa isyarat dengan menggunakan kombinasi fitur statis dengan dinamis dan metode L-GCNN memiliki akurasi 82,97%. Sedangkan menggunakan kombinasi fitur statis dengan dinamis dan metode RB-L-GCNN memiliki akurasi 87,04%. Hal ini dapat dibandingkan bahwa pengenalan bahasa isyarat dengan menggunakan kombinasi fitur statis dengan dinamis dan metode RB-L-GCNN memiliki peningkatan akurasi 4,07% lebih baik dari L-GCNN.

Begitu juga perbandingan dari penggunaan fitur statis dan kombinasi dari fitur statis dengan fitur dinamis. Penggunaan fitur statis dan metode L-GCNN memiliki akurasi 76,30% dan penggunaan kombinasi fitur statis dengan fitur dinamis dan metode L-GCNN memiliki akurasi 82,97%. Penggunaan kombinasi fitur statis dengan fitur dinamis dan L-GCNN memiliki peningkatan akurasi 6,67% lebih baik dibandingkan dengan penggunaan fitur statis.

Pada pengenalan bahasa isyarat dengan menggunakan fitur statis dan RB-L-GCNN memiliki akurasi 78,52%. Sedangkan penggunaan kombinasi fitur statis dengan fitur dinamis dan RB-L-GCNN memiliki akurasi 87,04%. Penggunaan kombinasi fitur statis dengan fitur dinamis memiliki peningkatan akurasi 8,52% lebih baik dibandingkan dengan penggunaan fitur statis.

4.4 Pembahasan Hasil

Pengenalan bahasa isyarat SIBI dengan menggunakan fitur statis dan RB-L-GCNN memiliki akurasi lebih baik dibandingkan dengan menggunakan metode

L-GCNN, *Decision tree*, SVM dan KNN. Begitu juga ketika menggunakan kombinasi fitur statis dengan fitur dinamis, metode RB-L-GCNN memiliki akurasi yang lebih baik dibandingkan dengan metode L-GCNN, *Decision tree*, SVM dan KNN.

Dari hasil skenario uji coba A penggunaan fitur statis memiliki akurasi lebih baik dibandingkan dengan menggunakan kombinasi fitur statis dengan fitur dinamis. Namun pada hasil skenario uji coba B, kombinasi fitur statis dengan fitur dinamis memiliki akurasi lebih baik dibandingkan dengan hanya menggunakan fitur statis. Pada skenario uji coba A, kombinasi fitur statis dengan fitur dinamis memiliki akurasi lebih rendah karena pengambilan data yang kurang tepat atau kesalahan disaat perekaman data. Pada skenario uji coba B, kombinasi fitur statis dengan fitur dinamis dapat meningkatkan akurasi pengenalan karena dengan menggunakan kombinasi fitur tersebut dapat mengenal bahasa isyarat yang bersifat statis maupun bahasa isyarat yang bersifat dinamis. Selain itu, uji coba yang dilakukan secara *real time* dapat menyesuaikan posisi tangan dengan baik dibandingkan dengan uji coba manual.

Kombinasi fitur statis dengan fitur dinamis dapat mengenal bahasa isyarat yang bersifat statis dan dinamis dengan baik. Dari hasil uji coba yang dilakukan, untuk pengenalan huruf yang bersifat dinamis seperti huruf J dan Z dapat dikenal dengan baik. Ada beberapa huruf statis yang tidak dapat dikenal dengan baik, karena beberapa huruf tersebut memiliki tingkat kemiripan yang tinggi. Adapun huruf yang memiliki kemiripan pertama adalah huruf A, M, N, O, dan S. Untuk huruf yang memiliki tingkat kemiripan kedua adalah I dan Y. Huruf-huruf yang memiliki tingkat kemiripan ketiga adalah K, P, dan V.

LMC memiliki kelemahan deteksi ketika tangan tidak dalam kondisi bentuk yang sempurna. Ketika tangan dalam bentuk huruf M dan N, LMC tidak dapat mendeteksi posisi titik-titik koordinat pada jari jempol dengan sempurna karena jempol akan selalu tertutup oleh jari lainnya. Hal ini terjadi karena LMC bekerja dengan menggunakan sensor infrared untuk mendeteksi keberadaan objek, sehingga ketika objek tertutup oleh objek lainnya maka objek tersebut tidak dapat dideteksi oleh sensor yang dimiliki oleh LMC.

LAMPIRAN 1

Data ke-	Fitur										Kelas
	Fitur statis			Fitur dinamis							
	Average spread	Average tri-spread	Extended distance	Frame ke-							
				1	2	3	4	5	6	7	8	9	
1	47.08	2359.94	100.79	0	0	0	0	0	0	0	0	0	NR
2	52.17	2727.52	100.73	0	0	0	0	0	0	0	0	0	NR
3	54.22	2865.8	101.14	0	0	0	0	0	0	0	0	0	NR
4	48.29	2448.08	98.89	0	0	0	0	0	0	0	0	0	NR
5	51.39	2676.2	101.85	0	0	0	0	0	0	0	0	0	NR
6	49.74	2575.72	101.09	0	0	0	0	0	0	0	0	0	NR
7	49.6	2499.22	92.27	0	0	0	0	0	0	0	0	0	NR
8	51.39	2620.91	94.89	0	0	0	0	0	0	0	0	0	NR
9	49.56	2485.39	96.85	0	0	0	0	0	0	0	0	0	NR
10	51.83	2637.08	93.71	0	0	0	0	0	0	0	0	0	NR
11	24.25	808.5	59.07	0	0	0	0	0	0	0	0	0	A
12	24.04	740.86	55.8	0	0	0	0	0	0	0	0	0	A
13	23.34	747.05	56.23	0	0	0	0	0	0	0	0	0	A
14	24.8	822.08	55.21	0	0	0	0	0	0	0	0	0	A
15	23.02	742.15	55.55	0	0	0	0	0	0	0	0	0	A
16	24.07	799.97	58.07	0	0	0	0	0	0	0	0	0	A
17	32.55	744.77	57.67	0	0	0	0	0	0	0	0	0	A
18	23.22	730.74	56.79	0	0	0	0	0	0	0	0	0	A
19	24.9	795.43	58.68	0	0	0	0	0	0	0	0	0	A
20	24.66	797.17	55.1	0	0	0	0	0	0	0	0	0	A
21	36.52	1489.3	89.4	0	0	0	0	0	0	0	0	0	B
22	36.91	1456.75	90.3	0	0	0	0	0	0	0	0	0	B
23	38.36	1592.59	90.5	0	0	0	0	0	0	0	0	0	B
24	38.41	1581.65	89.29	0	0	0	0	0	0	0	0	0	B
25	37.29	1520.55	97.19	0	0	0	0	0	0	0	0	0	B
26	37.44	1532.36	96.07	0	0	0	0	0	0	0	0	0	B
27	37.96	1563.29	95.01	0	0	0	0	0	0	0	0	0	B
28	38.69	1618.74	96.38	0	0	0	0	0	0	0	0	0	B
29	36.55	1462.65	94.18	0	0	0	0	0	0	0	0	0	B
30	37.67	1548.16	96.64	0	0	0	0	0	0	0	0	0	B
31	31.52	1349.71	79.24	0	0	0	0	0	0	0	0	0	C
32	32.66	1471.69	76.43	0	0	0	0	0	0	0	0	0	C
33	30.79	1347.33	76.39	0	0	0	0	0	0	0	0	0	C
34	31.37	1392.82	82.53	0	0	0	0	0	0	0	0	0	C
35	36.64	1663.31	80.25	0	0	0	0	0	0	0	0	0	C

36	36.9	1432.61	80.13	0	0	0	0	0	0	0	0	0	C
37	31.51	1392.95	78.55	0	0	0	0	0	0	0	0	0	C
38	31.73	1424.03	78.32	0	0	0	0	0	0	0	0	0	C
39	32.01	1377.58	74.35	0	0	0	0	0	0	0	0	0	C
40	35.78	1458.03	80.02	0	0	0	0	0	0	0	0	0	C
41	56.44	2659.46	88.35	0	0	0	0	0	0	0	0	0	D
42	58.63	2543.81	94.18	0	0	0	0	0	0	0	0	0	D
43	55.87	2529.34	93.16	0	0	0	0	0	0	0	0	0	D
44	58.18	2282.41	92.12	0	0	0	0	0	0	0	0	0	D
45	58.95	2327.49	90.15	0	0	0	0	0	0	0	0	0	D
46	56.4	2420.98	90.76	0	0	0	0	0	0	0	0	0	D
47	57.4	2388.55	90.71	0	0	0	0	0	0	0	0	0	D
48	57.14	2422.43	92.97	0	0	0	0	0	0	0	0	0	D
49	56.87	2377.32	90.49	0	0	0	0	0	0	0	0	0	D
50	56.61	2492.09	98.34	0	0	0	0	0	0	0	0	0	D
51	23.18	896.73	71.33	0	0	0	0	0	0	0	0	0	E
52	22.46	930.63	63.21	0	0	0	0	0	0	0	0	0	E
53	23.48	944.46	61.59	0	0	0	0	0	0	0	0	0	E
54	23.21	959.5	62.2	0	0	0	0	0	0	0	0	0	E
55	22.2	837.39	58.93	0	0	0	0	0	0	0	0	0	E
56	23.56	899.72	60.8	0	0	0	0	0	0	0	0	0	E
57	22.13	833.8	59.7	0	0	0	0	0	0	0	0	0	E
58	22.59	815.58	63.93	0	0	0	0	0	0	0	0	0	E
59	26.43	929.55	65.89	0	0	0	0	0	0	0	0	0	E
60	25.15	852.93	65.96	0	0	0	0	0	0	0	0	0	E
61	39.2	1941.02	98.19	0	0	0	0	0	0	0	0	0	F
62	4014	1893.55	92.11	0	0	0	0	0	0	0	0	0	F
63	40.3	1966.52	94.41	0	0	0	0	0	0	0	0	0	F
64	39.68	1972.98	97.66	0	0	0	0	0	0	0	0	0	F
65	39.72	1977.6	98.35	0	0	0	0	0	0	0	0	0	F
66	37.63	1841.2	98.61	0	0	0	0	0	0	0	0	0	F
67	36	1737.41	93.73	0	0	0	0	0	0	0	0	0	F
68	37.68	1814.41	92.31	0	0	0	0	0	0	0	0	0	F
69	36.27	1712.83	101.3	0	0	0	0	0	0	0	0	0	F
70	38.21	1824.41	107.32	0	0	0	0	0	0	0	0	0	F
71	34.84	1135.79	80.7	0	0	0	0	0	0	0	0	0	G
72	33.36	969.29	77.19	0	0	0	0	0	0	0	0	0	G
73	32.48	941.18	77.42	0	0	0	0	0	0	0	0	0	G
74	35.73	1120.85	75.72	0	0	0	0	0	0	0	0	0	G
75	35.74	1072.76	75.86	0	0	0	0	0	0	0	0	0	G
76	37.04	1191.25	75.06	0	0	0	0	0	0	0	0	0	G
77	33.67	1020.44	79.17	0	0	0	0	0	0	0	0	0	G

78	34.01	960.83	80.54	0	0	0	0	0	0	0	0	0	G
79	33.71	1025.28	82.56	0	0	0	0	0	0	0	0	0	G
80	36	990.81	81.72	0	0	0	0	0	0	0	0	0	G
81	54.5	2098.43	90.04	0	0	0	0	0	0	0	0	0	H
82	54.26	2109.55	96.84	0	0	0	0	0	0	0	0	0	H
83	54.81	1995.64	93.89	0	0	0	0	0	0	0	0	0	H
84	51.43	1964.61	96.98	0	0	0	0	0	0	0	0	0	H
85	53.23	1991.03	97.3	0	0	0	0	0	0	0	0	0	H
86	54.43	2102.84	97.39	0	0	0	0	0	0	0	0	0	H
87	54.49	2133.71	97.69	0	0	0	0	0	0	0	0	0	H
88	52.99	1973.19	91.51	0	0	0	0	0	0	0	0	0	H
89	55.17	2144.46	98.33	0	0	0	0	0	0	0	0	0	H
90	55.44	2131.85	98.01	0	0	0	0	0	0	0	0	0	H
91	43.2	1327.01	75.96	0	0	0	0	0	0	0	0	0	I
92	43.89	1429.72	74.86	0	0	0	0	0	0	0	0	0	I
93	40.77	1339.63	79.2	0	0	0	0	0	0	0	0	0	I
94	35.83	1167.02	90.14	0	0	0	0	0	0	0	0	0	I
95	39.83	1298.11	77.59	0	0	0	0	0	0	0	0	0	I
96	35.46	1174.65	81.51	0	0	0	0	0	0	0	0	0	I
97	31.66	1168.84	72.08	0	0	0	0	0	0	0	0	0	I
98	34.57	1201.02	75.24	0	0	0	0	0	0	0	0	0	I
99	35.37	1203.85	81.37	0	0	0	0	0	0	0	0	0	I
100	36.32	1147.92	77.44	0	0	0	0	0	0	0	0	0	I
101	37.25	1328.68	217.35	0	0	0	3	3	4	4	4	5	J
102	34.75	1111.25	218.15	0	0	0	3	3	4	0	4	0	J
103	36.38	1180.24	226.76	0	0	3	2	3	4	4	4	4	J
104	37.61	1226.56	245.41	0	0	3	0	0	0	2	4	4	J
105	36.31	1199.55	274.52	0	0	3	0	0	3	4	4	3	J
106	28.24	961.82	213.14	0	0	3	3	3	4	4	4	0	J
107	32.73	1048.38	240.83	0	0	3	3	3	4	4	4	0	J
108	33.02	1043.65	222.8	0	0	3	0	0	0	4	4	0	J
109	37.71	1277.04	191.07	0	0	0	3	3	3	4	4	2	J
110	35.49	1003.75	234.89	0	0	0	3	3	4	4	4	7	J
111	70.17	2889.26	100.06	0	0	0	0	0	0	0	0	0	K
112	68.2	2761.4	101.78	0	0	0	0	0	0	0	0	0	K
113	71.03	2892.97	99.63	0	0	0	0	0	0	0	0	0	K
114	70.21	2886.75	106	0	0	0	0	0	0	0	0	0	K
115	69.58	2834.67	99.33	0	0	0	0	0	0	0	0	0	K
116	68.89	2799.93	101.26	0	0	0	0	0	0	0	0	0	K
117	71.31	2876.92	100.82	0	0	0	0	0	0	0	0	0	K
118	67.92	2988.45	105.58	0	0	0	0	0	0	0	0	0	K
119	66.43	2595.67	100.49	0	0	0	0	0	0	0	0	0	K

120	73.31	2970.58	101.03	0	0	0	0	0	0	0	0	0	K
121	64.68	2576.96	92.05	0	0	0	0	0	0	0	0	0	L
122	64.41	2502.3	94.3	0	0	0	0	0	0	0	0	0	L
123	65.42	2584.52	96.53	0	0	0	0	0	0	0	0	0	L
124	64.21	2494.44	92.36	0	0	0	0	0	0	0	0	0	L
125	63.89	2501.47	96.08	0	0	0	0	0	0	0	0	0	L
126	62.76	2428.07	94.75	0	0	0	0	0	0	0	0	0	L
127	63.58	2483.42	96.18	0	0	0	0	0	0	0	0	0	L
128	63.56	2437.65	93.24	0	0	0	0	0	0	0	0	0	L
129	62.41	2372.88	93.38	0	0	0	0	0	0	0	0	0	L
130	60.72	2277.19	139.09	0	0	0	0	0	0	0	0	0	L
131	19.32	650.8	61.58	0	0	0	0	0	0	0	0	0	M
132	17.11	591.19	58.9	0	0	0	0	0	0	0	0	0	M
133	20.17	647.21	60.67	0	0	0	0	0	0	0	0	0	M
134	21.93	696.66	64.23	0	0	0	0	0	0	0	0	0	M
135	15.29	572.85	61.77	0	0	0	0	0	0	0	0	0	M
136	18.71	600.15	63.8	0	0	0	0	0	0	0	0	0	M
137	16.56	572.67	64.09	0	0	0	0	0	0	0	0	0	M
138	19.28	609.89	64.53	0	0	0	0	0	0	0	0	0	M
139	16.79	591.78	61.41	0	0	0	0	0	0	0	0	0	M
140	19.99	719.83	59.05	0	0	0	0	0	0	0	0	0	M
141	18.4	510.51	61.93	0	0	0	0	0	0	0	0	0	N
142	14.97	429.87	56.76	0	0	0	0	0	0	0	0	0	N
143	15.31	471.62	74.94	0	0	0	0	0	0	0	0	0	N
144	15.82	444.67	58.01	0	0	0	0	0	0	0	0	0	N
145	15.56	483.9	55.01	0	0	0	0	0	0	0	0	0	N
146	19.23	623.71	63.49	0	0	0	0	0	0	0	0	0	N
147	17.32	520.49	56.56	0	0	0	0	0	0	0	0	0	N
148	14.68	446.5	66.23	0	0	0	0	0	0	0	0	0	N
149	17.47	510.85	72.32	0	0	0	0	0	0	0	0	0	N
150	20.05	620.47	56.57	0	0	0	0	0	0	0	0	0	N
151	21.91	879.8	62.51	0	0	0	0	0	0	0	0	0	O
152	20.64	809.25	60.14	0	0	0	0	0	0	0	0	0	O
153	25.08	987.5	70.05	0	0	0	0	0	0	0	0	0	O
154	21.48	862.48	62.04	0	0	0	0	0	0	0	0	0	O
155	20.47	813.38	59.35	0	0	0	0	0	0	0	0	0	O
156	19.59	760.97	58.25	0	0	0	0	0	0	0	0	0	O
157	19.73	759.21	60.7	0	0	0	0	0	0	0	0	0	O
158	23.21	952.7	64.94	0	0	0	0	0	0	0	0	0	O
159	19.59	773.78	59.53	0	0	0	0	0	0	0	0	0	O
160	20.22	804.65	59.83	0	0	0	0	0	0	0	0	0	O
161	49.02	1596.38	88.09	0	0	0	0	0	0	0	0	0	P

162	48.71	1664.28	85.38	0	0	0	0	0	0	0	0	0	P
163	50.02	1685.46	84.75	0	0	0	0	0	0	0	0	0	P
164	44.61	1417.29	87.16	0	0	0	0	0	0	0	0	0	P
165	44.81	1536.22	94.32	0	0	0	0	0	0	0	0	0	P
166	49.83	1707.3	84.75	0	0	0	0	0	0	0	0	0	P
167	46.24	1548.8	87.43	0	0	0	0	0	0	0	0	0	P
168	46.49	1548.96	88.18	0	0	0	0	0	0	0	0	0	P
169	53.04	1819.11	89.29	0	0	0	0	0	0	0	0	0	P
170	50.27	1698.89	88.53	0	0	0	0	0	0	0	0	0	P
171	42.92	1374.64	82.01	0	0	0	0	0	0	0	0	0	Q
172	36.37	1064.29	78.14	0	0	0	0	0	0	0	0	0	Q
173	40.36	1265.87	80.99	0	0	0	0	0	0	0	0	0	Q
174	38.06	1340.27	76.11	0	0	0	0	0	0	0	0	0	Q
175	40.3	1299.92	81	0	0	0	0	0	0	0	0	0	Q
176	37.95	1132.64	84.95	0	0	0	0	0	0	0	0	0	Q
177	37.3	1093.17	83.75	0	0	0	0	0	0	0	0	0	Q
178	36.53	1040.6	83.79	0	0	0	0	0	0	0	0	0	Q
179	38.54	1284.4	81.32	0	0	0	0	0	0	0	0	0	Q
180	38.6	1231.88	80.09	0	0	0	0	0	0	0	0	0	Q
181	52.28	1819.29	94.95	0	0	0	0	0	0	0	0	0	R
182	50.27	1632.57	97.85	0	0	0	0	0	0	0	0	0	R
183	55.46	2239.7	98.15	0	0	0	0	0	0	0	0	0	R
184	52.27	2009.93	96.03	0	0	0	0	0	0	0	0	0	R
185	55.46	2000.02	100.37	0	0	0	0	0	0	0	0	0	R
186	53.65	1766.54	100.22	0	0	0	0	0	0	0	0	0	R
187	51.77	1660.78	97.97	0	0	0	0	0	0	0	0	0	R
188	58.64	2030.53	99.6	0	0	0	0	0	0	0	0	0	R
189	57.75	2013.16	101.1	0	0	0	0	0	0	0	0	0	R
190	56.75	1977.26	99.27	0	0	0	0	0	0	0	0	0	R
191	19.44	656.28	65.56	0	0	0	0	0	0	0	0	0	S
192	21.07	710.02	69.68	0	0	0	0	0	0	0	0	0	S
193	21.17	655.47	59.37	0	0	0	0	0	0	0	0	0	S
194	20.48	693.42	69.57	0	0	0	0	0	0	0	0	0	S
195	13.72	384.74	56.96	0	0	0	0	0	0	0	0	0	S
196	16.65	566.08	60.3	0	0	0	0	0	0	0	0	0	S
197	15.26	479.25	59.98	0	0	0	0	0	0	0	0	0	S
198	14.22	427.68	55.65	0	0	0	0	0	0	0	0	0	S
199	16.32	497.13	61.17	0	0	0	0	0	0	0	0	0	S
200	16.19	468.78	55.02	0	0	0	0	0	0	0	0	0	S
201	38.63	1022.44	81.06	0	0	0	0	0	0	0	0	0	T
202	39.74	1064.75	75.95	0	0	0	0	0	0	0	0	0	T
203	37.18	1071.47	77.49	0	0	0	0	0	0	0	0	0	T

204	43.88	1177.48	85.32	0	0	0	0	0	0	0	0	0	T
205	43.21	1144.75	71.75	0	0	0	0	0	0	0	0	0	T
206	40.84	1088.48	84.1	0	0	0	0	0	0	0	0	0	T
207	40.78	1089.76	82.66	0	0	0	0	0	0	0	0	0	T
208	42.74	1228.28	80.54	0	0	0	0	0	0	0	0	0	T
209	42.39	1281.06	77.98	0	0	0	0	0	0	0	0	0	T
210	41.46	1242.41	76.83	0	0	0	0	0	0	0	0	0	T
211	51.42	1969.6	99.67	0	0	0	0	0	0	0	0	0	U
212	53.79	2143.17	102.02	0	0	0	0	0	0	0	0	0	U
213	50.79	1912.56	111.75	0	0	0	0	0	0	0	0	0	U
214	55.9	2220.45	99.83	0	0	0	0	0	0	0	0	0	U
215	51.39	1867.83	100.39	0	0	0	0	0	0	0	0	0	U
216	54.58	2169.89	101.13	0	0	0	0	0	0	0	0	0	U
217	53.79	2076.29	101.2	0	0	0	0	0	0	0	0	0	U
218	53.98	2056.37	97.04	0	0	0	0	0	0	0	0	0	U
219	54.34	2052.74	100.85	0	0	0	0	0	0	0	0	0	U
220	51.05	1839.51	106.97	0	0	0	0	0	0	0	0	0	U
221	75.36	3345.77	100.43	0	0	0	0	0	0	0	0	0	V
222	76.01	3294.71	100.05	0	0	0	0	0	0	0	0	0	V
223	73.85	3317.24	100.82	0	0	0	0	0	0	0	0	0	V
224	74.07	3325	100.71	0	0	0	0	0	0	0	0	0	V
225	72.43	3092.74	102.73	0	0	0	0	0	0	0	0	0	V
226	73.86	3392.36	99.88	0	0	0	0	0	0	0	0	0	V
227	65.79	3052.93	99.08	0	0	0	0	0	0	0	0	0	V
228	73.79	3461.21	98.32	0	0	0	0	0	0	0	0	0	V
229	71.9	3324.41	101.08	0	0	0	0	0	0	0	0	0	V
230	64.59	3010.93	82.91	0	0	0	0	0	0	0	0	0	V
231	54.93	2262.11	88.54	0	0	0	0	0	0	0	0	0	W
232	54.63	2274.35	92.14	0	0	0	0	0	0	0	0	0	W
233	54.9	2293.49	90.98	0	0	0	0	0	0	0	0	0	W
234	51.77	2112.34	87.85	0	0	0	0	0	0	0	0	0	W
235	53.17	2243.59	88.01	0	0	0	0	0	0	0	0	0	W
236	56.37	2248.2	89.62	0	0	0	0	0	0	0	0	0	W
237	55.13	2302.51	88.48	0	0	0	0	0	0	0	0	0	W
238	53.07	2224.1	87.16	0	0	0	0	0	0	0	0	0	W
239	56.8	2452.05	87.85	0	0	0	0	0	0	0	0	0	W
240	56.24	2322.66	88.24	0	0	0	0	0	0	0	0	0	W
241	51.6	1790.46	76.35	0	0	0	0	0	0	0	0	0	X
242	49.87	1670.33	79.48	0	0	0	0	0	0	0	0	0	X
243	46.31	1511.46	81.56	0	0	0	0	0	0	0	0	0	X
244	46.9	1726.85	85.54	0	0	0	0	0	0	0	0	0	X
245	47.07	1758.97	81.22	0	0	0	0	0	0	0	0	0	X

246	48	1698.36	86.8	0	0	0	0	0	0	0	0	0	X
247	48.42	1771.26	76.72	0	0	0	0	0	0	0	0	0	X
248	45.64	1585.26	83.77	0	0	0	0	0	0	0	0	0	X
249	46.77	1654.48	71.26	0	0	0	0	0	0	0	0	0	X
250	46.05	1506.27	84.71	0	0	0	0	0	0	0	0	0	X
251	43.06	1295.38	72.72	0	0	0	0	0	0	0	0	0	Y
252	44.17	1316.37	70.98	0	0	0	0	0	0	0	0	0	Y
253	38.63	1144.78	68.93	0	0	0	0	0	0	0	0	0	Y
254	33.24	953.34	68.83	0	0	0	0	0	0	0	0	0	Y
255	33.11	993.36	71.17	0	0	0	0	0	0	0	0	0	Y
256	39.89	1136.14	74.44	0	0	0	0	0	0	0	0	0	Y
257	40.26	1167.79	68.01	0	0	0	0	0	0	0	0	0	Y
258	40.84	1199.67	72.43	0	0	0	0	0	0	0	0	0	Y
259	39.9	1153.05	68.66	0	0	0	0	0	0	0	0	0	Y
260	40.17	1153.21	77.97	0	0	0	0	0	0	0	0	0	Y
261	53.84	2033.77	197.95	0	0	0	1	3	3	0	0	0	Z
262	55.04	2005.41	192.91	0	0	0	3	3	3	0	0	0	Z
263	52.71	1876.28	188.53	0	0	0	0	3	3	0	0	0	Z
264	53.76	1928.57	171.25	0	0	0	1	3	3	0	0	0	Z
265	53.38	1903.69	188.85	0	0	0	0	3	3	3	0	0	Z
266	52.98	1822.79	145.85	0	0	1	3	3	3	0	0	0	Z
267	52.65	1840.15	179.21	0	0	0	3	3	0	0	0	0	Z
268	50.64	1758.4	159.7	0	0	0	3	3	3	0	0	0	Z
269	50.88	1724.81	185.67	0	0	1	3	3	0	0	0	5	Z
270	49.97	1802.74	166.53	0	0	0	0	3	3	0	0	0	Z

[*Halaman ini sengaja dikosongkan*]

LAMPIRAN 2

Daftar Singkatan :

ArSL : Arabic Sign Language
ASL : American Sign Language
BISINDO : Berkenalan dengan Sistem Isyarat Indonesia
FSL : French Sign Language
GERKATIN : Gerakan Kesejahteraan Tuna Rungu Indonesi
GCNN : General Classifier Neural Network
GRNN : Generalized Regression Neural Network
GSL : Germany Sign Language
HCI : Human Computer Interaction
KNN : K-Nearest Neighbor
L-GCNN : Logarithmic Learning for Generalized Classifier Neural Network
LMC : Leap Motion Controller
PNN : Probabilistic Neural Network
RB : Rule Based
SIBI : Sistem Isyarat Bahasa Indonesia
SVM : Support Vector Machine

[*Halaman ini sengaja dikosongkan*]

BAB 5
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang dapat diambil selama penulis melakukan penelitin ini serta saran-saran kedepan untuk meningkatkan penelitian ini.

5.1 Kesimpulan

Dari hasil pengujian dan analisa pada penelitian ini, dapat diambil beberapa kesimpulan sebagai berikut :

1. Dengan hanya menggunakan fitur statis pada pengenalan bahasa isyarat SIBI tidak dapat mengenal bahasa isyarat SIBI yang bersifat dinamis seperti huruf J dan Z dengan baik.
2. Dengan mengkombinasikan fitur statis dan fitur dinamis yang didapatkan dari perangkat keras LMC untuk pengenalan bahasa isyarat SIBI bahwa kombinasi fitur tersebut dapat mengenal huruf yang bersifat statis maupun huruf yang bersifat dinamis dengan baik.
3. Pembentukan *rule based* dengan menggunakan *decision tree* pada metode klasifikasi L-GCNN (RB-L-GCNN) dapat meningkatkan akurasi pengenalan bahasa isyarat dengan menggunakan fitur statis hingga 4.07% dan ketika menggunakan kombinasi fitur statis dan fitur dinamis hingga 6.67%.
4. Pembentukan *rule based* menggunakan *decision tree* pada L-GCNN dan kombinasi fitur statis dengan fitur dinamis dapat meningkatkan kualitas pengenalan bahasa isyarat SIBI.

5.2 Saran

Adapun saran penulis untuk pengembangan pengenalan bahasa isyarat SIBI kedepannya adalah :

1. Pembentukan fitur yang dapat membedakan beberapa huruf yang memiliki tingkat kemiripan yang sangat tinggi seperti huruf (A, E, M, N, S) dan (K dengan V).

2. Pembentukan *rule* yang lebih baik lagi yang dapat memisahkan fitur dari huruf-huruf yang serupa.
3. Perlu adanya penelitian lebih lanjut tentang kalibrasi dari jari tangan setiap pengguna, karena ukuran tangan setiap pengguna bebeda sehingga dapat mempengaruhi hasil fitur.

DAFTAR PUSTAKA

Chen, Y. et al., 2015. Rapid Recognition of Dynamic Hand Gestures using Leap Motion. , (August), pp.1419–1424.

Chuan, C. et al., 2014. American Sign Language Recognition Using Leap Motion Sensor. , pp.541–544.

Elons, a. S., Abull-Ela, M. & Tolba, M.F., 2013. A proposed PCNN features quality optimization technique for pose-invariant 3D Arabic sign language recognition. *Applied Soft Computing Journal*, 13(4), pp.1646–1660. Available at: http://dx.doi.org/10.1016/j.asoc.2012.11.036.

Elons, A.S. et al., 2014. Arabic Sign Language Recognition Using Leap Motion Sensor. , pp.368–373.

Khotimah, W.N. et al., 2015. Comparison between Back Propagation Neural Network and Genetic Algorithm Back Propagation Neural Network for Sign Language Recognition. , 00, pp.0–5.

Marin, G., Dominio, F. & Zanuttigh, P., 2014. Hand gesture recognition with leap motion and kinect devices. *2014 IEEE International Conference on Image Processing (ICIP)*, pp.1565–1569. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7025313.

Melis, B. & Avci, M., 2013. Generalized classifier neural network. *Neural Networks*, 39, pp.18–26.

Melis, B. & Avci, M., 2014. Logarithmic learning for generalized classifier neural network. *Neural Networks*, 60, pp.133–140.

Mohandes, M., Aliyu, S. & Deriche, M., 2014. Arabic Sign Language Recognition using the Leap Motion Controller.

Mohandes, M., Aliyu, S. & Deriche, M., 2015. Prototype Arabic Sign Language Recognition using Multi-Sensor Data Fusion of Two Leap Motion Controllers. , pp.1–6.

Oz, C. & Leu, M.C., 2007. Linguistic properties based on American Sign Language isolated word recognition with artificial neural networks using a sensory glove and motion tracker. *Neurocomputing*, 70(7), pp.2891–2901. Available at: http://dx.doi.org/10.1016/j.engappai.2011.06.015.

Sharma, R.P. & Verma, G.K., 2015. Human Computer Interaction using Hand Gesture. *Procedia - Procedia Computer Science*, 54, pp.721–727. Available at: http://dx.doi.org/10.1016/j.procs.2015.06.085.

Shukor, A.Z. et al., 2015. A New Data Glove Approach for Malaysian Sign Language Detection. *Procedia Computer Science*, 76(Iris), pp.60–67. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1877050915037771.

Sugianto, N. & Samopa, F., 2015. Analisa Manfaat Dan Penerimaan Terhadap Implementasi Bahasa Isyarat Indonesia Pada Latar Belakang Komplek Menggunakan Kinect Dan Jaringan Syaraf Tiruan (Studi Kasus SLB Karya Mulia 1). , 01(01), pp.56–72.

Zaki, M.M. & Shaheen, S.I., 2011. Sign language recognition using a combination of new vision based features. *Pattern Recognition Letters*, 32(4), pp.572–577. Available at: http://dx.doi.org/10.1016/j.patrec.2010.11.013.

Zhou, Y., Jiang, G. & Lin, Y., 2016. A novel fi nger and hand pose estimation technique for real-time hand gesture recognition. *Pattern Recognition*, 49, pp.102–114. Available at: http://dx.doi.org/10.1016/j.patcog.2015.07.014.

BIODATA PENULIS

Supria atau dengan nama panggilan **Pia**, lahir di Bengkalis, Riau pada tanggal 12 Agustus 1987, merupakan anak kedua dari tiga bersaudara. Penulis pertama kali menempuh sekolah dasar pada tahun 1994 di SDN 045 Bantan, selanjutnya pada tahun 2000 di MTS Ar-rasyidin Bantan, dan selanjutnya pada tahun 2003 di MA Al-Ulum Bantan, sehingga lulus pada tahun 2006. Pada tahun 2006 penulis mulai belajar di perguruan tinggi Politeknik Negeri Bengkalis sampai selesai pada tahun 2009. Pada tahun 2012 penulis melanjutkan pendidikan pada jalur alih jenjang D3 ke D4 jurusan Teknologi Media Digital (TMD) KERMA SEAMOLEC-ITB di Institut Teknologi Bandung sampai lulus pada tahun 2013. Kemudian pada tahun 2014 dengan izin dan rahmat Allah SWT, penulis di terima di Magister Teknik Informatika Fakultas Teknologi Informasi ITS dan berhasil menyelesaikan studi pada tahun 2016 dengan bidang minat Interaksi, Grafik dan Seni (IGS) dan Komputasi Cerdas dan Visualisasi (KCV). Penulis memiliki ketertarikan dalam bidang *Human Computer Interaction* (HCI), Grafika, *Image Processing* dan *Computer Vision*. Kontak penulis dengan email : phiya1287@gmail.com.

[*Halaman ini sengaja dikosongkan*]