

PENGENALAN SISTEM ISYARAT BAHASA INDONESIA MENGGUNAKAN KOMBINASI FITUR STATIS DAN FITUR DINAMIS LMC BERBASIS L-GCNN

Supria¹⁾, Darlis Heru Murti²⁾, dan Wijayanti Nurul Khotimah³⁾

¹⁾ Jurusan Teknik Informatika, Politeknik Negeri Bengkalis
Jl. Bathin Alam, Sungai Alam, Bengkalis, Riau, 28711

^{2, 3)} Jurusan Teknik Informatika, Institut Teknologi Sepuluh Nopember
Jl. Raya ITS, Kampus ITS, Sukolilo, Surabaya, 60111

e-mail: phiya1287@gmail.com¹⁾, darlis.herumurti@gmail.com²⁾, wijay.cs@gmail.com³⁾

ABSTRAK

Proses komunikasi antara penyandang tunarungu dan tunawicara dapat dipahami antara sesama dengan baik karena mereka sudah terbiasa sehari-harinya menggunakan bahasa isyarat. Namun sebagian orang normal akan kesulitan untuk memahami bahasa isyarat yang disampaikan oleh penyandang tunarungu dan tunawicara, begitu juga sebaliknya. Untuk mengatasi masalah tersebut maka dibangun sebuah sistem pengenalan bahasa isyarat dengan menggunakan leap motion controller (LMC). Pengenalan bahasa isyarat dengan hanya menggunakan fitur statis hanya dapat mengenal bahasa isyarat yang bersifat statis dan tidak dapat mengenal bahasa isyarat yang bersifat dinamis dengan baik.

Pada penelitian ini diusulkan pengenalan Sistem Isyarat Bahasa Indonesia (SIBI) dengan menggunakan kombinasi fitur statis dengan fitur dinamis berbasis Logarithmic Learning for Generalized Classifier Neural Network (L-GCNN). Dimana fitur statis dimanfaatkan untuk pengenalan bahasa isyarat yang bersifat statis dan fitur dinamis dimanfaatkan untuk mengenal bahasa isyarat yang bersifat dinamis. L-GCNN dimanfaatkan untuk meningkatkan akurasi pengenalan bahasa isyarat. Dari hasil pengujian yang dilakukan pengenalan bahasa isyarat SIBI dengan menggunakan kombinasi fitur statis dengan fitur dinamis dapat mengenal bahasa isyarat SIBI yang bersifat statis maupun bahasa isyarat yang bersifat dinamis dengan baik.

Kata Kunci: fitur statis dan dinamis, leap motion controller, L-GCNN, pengenalan bahasa isyarat.

ABSTRACT

Process of communication between the deaf and dumb people can be well understood by each other 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. To overcome these problems, we will develop a sign language recognition system uses Leap Motion Controller (LMC). The recognition of sign language by using only static features can only recognize static sign language and can not well recognize dynamic sign language.

In this study, we propose the SIBI sign language recognition which combining static and dynamic features of the LMC based on Logarithmic Learning for Generalized Classifier Neural Network (L-GCNN). The static features are used to recognize the static sign language, and the dynamic features are used to recognize the dynamic sign language. L-GCNN was used to improve the accuracy of recognition. From the results of test performed SIBI recognition using only static features can only recognize static sign language and can not well recognize dynamic sign language. While the SIBI recognition using combination of static features and dynamic features can recognize both static sign language and dynamic sign language well.

Keywords: leap motion controller, L-GCNN, sign language recognition, static and dynamic features.

I. PENDAHULUAN

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 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 ke dalam tiga kategori besar yaitu : berbasis *computer vision* [1], [2], [3], berbasis sensor *glove* [4], [5] dan sensor *motion*, dan kombinasi antara kedua metode tersebut [5]. Pengenalan bahasa isyarat berbasis komputer visi membutuhkan komputasi yang sangat kompleks karena memperhitungkan setiap piksel pada citra. *Human Computer Interaction* (HCI) telah memperkenalkan sistem yang efektif pada pengenalan *hand gesture* [6].

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 [7]. Sistem tersebut menghasilkan akurasi pengenalan cukup baik untuk pengenalan bahasa isyarat yang bersifat statis. Beberapa fitur statis dari LMC 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* [6]. Sistem tersebut memanfaatkan fitur *gesture* pada *leap motion* untuk pengenalan cepat *dynamic hand gesture*. Namun sistem tersebut hanya dapat mengenal gerakan tangan.

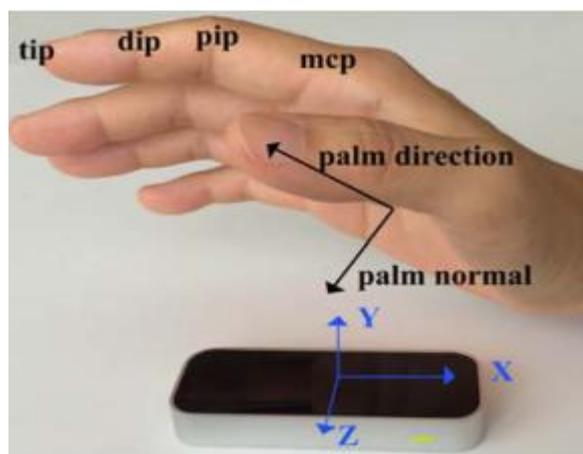
Generalized classifier neural network (GCNN) merupakan metode klasifikasi yang diusulkan untuk melakukan klasifikasi dengan akurasi yang tinggi [8]. Pada proses klasifikasi beberapa dataset yang diuji, GCNN memiliki akurasi yang lebih baik dibandingkan dengan metode GRNN dan PNN. *Logarithmic learning 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 [9]. 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.

Pada penelitian ini diusulkan sistem pengenalan bahasa isyarat SIBI yang mengkombinasikan fitur statis dan fitur dinamis yang didapatkan dari LMC berbasis L-GCNN. Fitur statis *average spread*, *average tri-spread*, dan *extended distance* dimanfaatkan untuk pengenalan bahasa isyarat yang bersifat statis sedangkan fitur dinamis *hand dynamic gesture* dimanfaatkan untuk pengenalan bahasa isyarat yang bersifat dinamis. L-GCNN digunakan untuk melakukan klasifikasi antara data uji coba terhadap data latih atau data model dari fitur statis dan fitur 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 penyandang tunarungu dan tunawicara.

II. TINJAUAN PUSTAKA

A. Bahasa Isyarat

Bahasa isyarat adalah salah satu bahasa komunikasi yang dilakukan dengan menggunakan pergerakan tangan, pergerakan tubuh, atau ekspresi 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) [7],[5],[10],[1], *French Sign Language* (FSL), *Germany Sign Language*



Gambar 1. Capture tangan menggunakan LMC

(GSL), dan *Arabic Sign Language* (ArSL) [11],[12],[13],[14].

Indonesia memiliki dua sistem bahasa isyarat nasional yaitu Berkenalan dengan Sistem Isyarat Indonesia (BISINDO) dan Sistem Isyarat Bahasa Indonesia (SIBI) [15]. BISINDO dikembangkan oleh orang penyandang tuna rungu itu sendiri melalui Gerakan Kesejahteraan Tuna Rungu Indonesia (GERKATIN), sedangkan SIBI dikembangkan oleh orang normal yang mengadopsi dari 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. Sistem 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).

B. Fitur Statis

Fitur statis adalah fitur yang bersifat statis yang didapatkan dari titik-titik koordinat tangan. Ada beberapa fitur statis yaitu:

1) Average distance

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 (1).

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

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

2) Average spread

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).

$$averageSpread = \frac{1}{T} \sum_{t=1}^T \sum_{n=1}^4 |tip_t^{n+1} - tip_t^n|, \quad (2)$$

3) Average Tri-spread

Average Tri-spread adalah area segitiga (*triArea*) antara dua *tip* yang berdekatan dan titik tengah (*midpoint*) antara dua posisi *metacarpal* (*mcp* dapat dilihat pada Gambar 1). 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 segitiga ditunjukkan pada Persamaan (3).

dimana θ adalah sudut antara dua vector.

Average tri-spread dapat dihitung dengan menambahkan area segitiga dari semua pasangan jari dan dibagi

$$\begin{aligned} triArea_{n,n+1} &= \frac{1}{2} \overrightarrow{tip_n mcp_{n,n+1}} \times \overrightarrow{tip_{n+1} mcp_{n,n+1}} \\ &= \frac{1}{2} \left| \overrightarrow{tip_n mcp_{n,n+1}} \right| \left| \overrightarrow{tip_{n+1} mcp_{n,n+1}} \right| \sin(\theta), \end{aligned} \quad (3)$$

dengan total dari jumlah frame. Adapun persamaan average spread ditunjukkan pada Persamaan (4).

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

4) *Extended distance*

Extended distance adalah jarak maksimum dari semua titik jari (*tip*, *dip*, *pip* dan *mcp* ditunjukkan pada Gambar 1) terhadap titik *palm center* (titik tengah tangan). Hasil nilai pada fitur ini memiliki satuan *millimeter* (mm).

C. *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* [6]. Pembentukan fitur yang baik akan meningkatkan akurasi dalam pengenalan. Lintasan gerakan diproyeksikan kedalam bidang *XOZ* yang merupakan prinsip pada bidang. Untuk setiap posisi titik *palm* $P_t(x_t, y_t, z_t)$ dalam urutan gerakan dinamis, orientasinya dalam bidang lintasan dapat digambarkan oleh vektor $\overrightarrow{P_{t-1} P_t}$ orientasi sudut mutlak dapat dicatat $\alpha_t \in (0, 360^\circ)$ dapat dilihat pada Persamaan (5) (6) dan (7).

$$\Delta z = z_t - z_{t-1} \tag{5}$$

$$\Delta x = x_t - x_{t-1} \tag{6}$$

$$\alpha_t = \begin{cases} \arctan\left(\frac{\Delta z}{\Delta x}\right) * \left(\frac{180}{\pi}\right) + 180, & \text{jika } \Delta x < 0, \\ \arctan\left(\frac{\Delta z}{\Delta x}\right) * \left(\frac{180}{\pi}\right) + 360, & \text{jika } \Delta z < 0, \\ \arctan\left(\frac{\Delta z}{\Delta x}\right) * \left(\frac{180}{\pi}\right), & \text{jika } \Delta x > 0 \text{ dan } \Delta z \geq 0, \end{cases} \tag{7}$$

dimana :

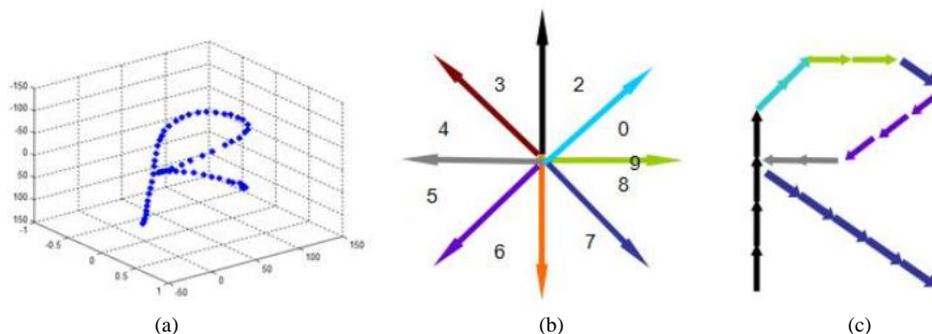
Δz = nilai selisih antara dua vector *z*,

Δx = nilai selisih antara dua vector *x*.

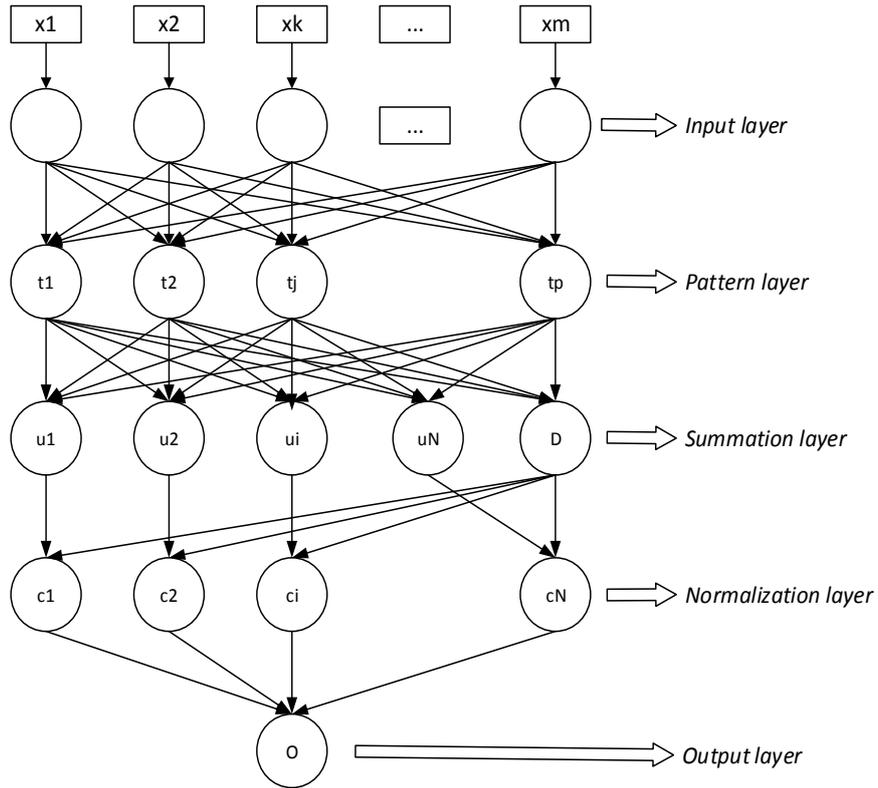
Setelah mencapai urutan orientasi sudut, untuk meningkatkan tingkat pengenalan, sudut α_t dikuantisasi dengan membaginya dengan 45° dan diberikan kode acuan dari 1 sampai 9 (9 merepresentasikan 0°). Seperti ditunjukkan pada Gambar 2 (a) adalah contoh pergerakan dalam bidang *XOZ*, sedangkan Gambar 2 (b) adalah acuan kuantisasi dari orientasi 360° , dan Gambar 2 (c) adalah contoh hasil kuantisasi untuk huruf R.

D. *Normalisasi*

Normalisasi data fitur (*feature scaling*) dilakukan untuk merubah data fitur kedalam *range* tertentu sehingga data fitur akan lebih proporsional [16]. 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



Gambar 2. Proses kuantisasi (A) Contoh pergerakan dalam bidang *XOZ*, (B) Acuan kuantisasi, (C) Hasil kuantisasi.



Gambar 3. Arsitektur L-GCNN.

menggunakan Persamaan (8).

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (8)$$

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

L-GCNN tidak seperti jaringan syaraf tiruan berbasis radial basis function lainnya [9]. Arsitektur L-GCNN dapat dilihat pada Gambar 3. 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 *output layer*.

Input layer mengirimkan vektor input x terpilih menuju ke *pattern layer*. *Pattern layer* terdiri atas satu neuron untuk tiap training datum. Neuron pada *pattern layer* digunakan untuk menghitung jarak *euclidian* antara vektor input x dan vektor *training data* t dengan menggunakan Persamaan (9). Dengan P menunjukkan jumlah dari data training.

$$dist(j) = \|x - t_j\|^2, 1 \leq j \leq P \quad (9)$$

Dengan:

x = Vector input x ,

t = Training data vector t ,

P = Menunjukkan jumlah dari *training data*.

Output pada *pattern layer* ditentukan dengan menggunakan fungsi aktivasi RBF. Seperti yang ditunjukkan pada Persamaan (10).

$$r(j) = e^{-1 * \frac{dist(j)}{2\sigma^2}}, 1 \leq j \leq P \quad (10)$$

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

Sebagai metodologi L-GCNN yang berbasis regresi, maka L-GCNN dibuat dengan struktur *one vs all discriminative*. Oleh karena itu, setiap *training datum* memiliki N nilai yang ditentukan dengan menentukan apakah data tersebut termasuk ke dalam suatu kelas atau tidak. Jika *training datum* termasuk ke dalam i^{th} class maka class ke i tersebut akan bernilai 0.9 dan class yang lainnya bernilai 0.1.

$$y(j, i) = \begin{cases} 0.9, & t_j \text{ termasuk kelas } i^{th} \ 1 \leq i \leq N \\ 0.1, & \text{else } 1 \leq j \leq p \end{cases} \quad (11)$$

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 $y(j, i) - y_{max}$, Persamaan (12) untuk meningkatkan efek dari $y(j, i)$. Maksud digunakannya fungsi eksponensial adalah untuk menyediakan titik temu (konvergen) 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.

$$d(j, i) = e^{(y(j,i)-y_{max})} * y(j, i) \quad (12)$$

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

Pada layer ini, N neuron menghitung jumlah dot produk dari *diverge effect term* dan *pattern layer outputs* seperti yang diberikan pada Persamaan (13).

$$u_i = \sum_{j=1}^p d(j, i) * (r(j)), 1 \leq i \leq N \quad (13)$$

dimana :

$d(j, i)$ = *Diverge effect*,
 $r(j)$ = *Output pattern layer*,
 N = Banyaknya *neuron*.

Kemudian neuron yang lainnya menghitung *denominator* seperti yang ditunjukkan pada Persamaan (14).

$$D = \sum_{j=1}^p r(j) \quad (14)$$

Dengan :

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

Pada *Normalization layer*, terdapat N neuron yang ditampilkan, tiap class dan *output* dari neuron tersebut dihitung dengan menggunakan Persamaan (15).

$$c_i = \frac{u_i}{D}, 1 \leq i \leq N \quad (15)$$

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 (16) dengan memilih nilai maksimum dari *output* dari *normalization layer*.

$$[o, id] = \max(c) \quad (16)$$

dengan :

o = nilai,
 id = kelas,
 $Max(c)$ = nilai *output neuron* pada layer *normalization* tertinggi.

L-GCNN menggunakan metode pembelajaran *logarithmic cost function* untuk pengoptimalisasian *smoothing parameter*. Tidak seperti GCNN yang biasanya, *cost function* didefinisikan dengan menggunakan fungsi *logarithmic* bukan error kuadrat seperti yang ditunjukkan pada Persamaan (17), dimana e menunjukkan *cost function*, $y(z, id)$ menunjukkan nilai dari data *training input* ke z^{th} untuk i^{th} 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 aktivasi 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.

$$e = (y(z, id) * \log(c_{id})) + (1 - y(z, id)) * \log(1 - c_{id}) \quad (17)$$

dengan :

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

Saat bagian pertama dari Persamaan (17) menunjukkan kuantitas *convergence* dengan nilai target sebesar 0.9 jika datum termasuk kedalam salah satu *class* kemudian bagian kedua dari persamaan menunjukkan kuantitas *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 (17) dan jika nilai target adalah 0.1 maka bagian kedua dari *cost function* akan sangat berpengaruh.

Smoothing parameter diperbaharui dengan menggunakan *gradient* dari fungsi biaya (e) dengan menggunakan Persamaan (18 - 22), dimana lr adalah *learning rate*.

$$\sigma_{new} = \sigma_{old} + lr * \frac{\partial e}{\partial \sigma} \quad (18)$$

$$\frac{\partial e}{\partial \sigma} = y(z, id) * \left(\frac{\frac{\partial c_{id}}{\partial \sigma}}{c_{id}} \right) + (1 - y(z, id)) * \left(\frac{\frac{\partial c_{id}}{\partial \sigma}}{c_{id}} \right) \quad (19)$$

$$\frac{\partial c_{id}}{\partial \sigma} = \frac{b(id) - l(id) * c_{id}}{D} \quad (20)$$

$$b(id) = 2 * \sum_{j=1}^p d(j, i) * (r(j), * \frac{dist(j)}{\sigma^3}) \quad (21)$$

$$l(id) = 2 * \sum_{j=1}^p (r(j), * \frac{dist(j)}{\sigma^3}) \quad (22)$$

dengan :

lr = *learning rate*,

- $dist(j)$ = jarak *euclidean*,
- $d(j, i)$ = *Diverge effect*,
- $r(j)$ = *Output pattern layer*,
- $y(z, id)$ = nilai dari data *training input* ke *zth* untuk *idth class*,
- 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 konvergen 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 memeriksa nilai baru dari *smoothing parameter* berlawanan dengan *constraint*-nya sebelum memperbaruinya. Jika berukuran dibawah nol maka *smoothing parameter* tidak akan *diupdate*.

Algoritma dari langkah-langkah pelatihan L-GCNN dapat dilihat pada Algoritma 1 dimana *epoch* menunjukkan banyaknya jumlah iterasi yang digunakan pada algoritma pelatihan, te menunjukkan *error total* yang dihitung terhadap keseluruhan *training data* pada iterasi sebelumnya dan a_{te} menunjukkan *total error* yang diterima. Ketika salah satu kriteria penghentian tercapai, maka pelatihan dihentikan dan *smoothing parameter* yang dapat diterima sebagai nilai *smoothing parameter* yang optimal di bawah toleransi kesalahan dan jumlah maksimum iterasi dicapai.

Algoritma 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 \leq epoch

for setiap data uji coba t_j

if iterasi > 1

if $(\sigma_j + lr * \frac{\partial e}{\partial \sigma_{id}}) > 0$

 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 ; $c_{max}(\text{iterasi}) = c_{id}$

 Hitung logarithmic cost

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

 dimana z menunjukkan input ke- z

end for

$y_{max} = \max(c_{max})$

 increment iterasi

if $te \leq a_{te}$

 Stop training

end while

Algoritma 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, $\text{dist}(j)$

 Tentukan dengan fungsi aktivasi 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

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 1. Sedangkan untuk proses uji coba dapat dilihat pada Algoritma 2.

III. METODOLOGI

Secara garis besar ada dua alur pada penelitian ini yaitu proses pelatihan dan proses uji coba. Pada proses pelatihan dan uji coba memiliki tiga tahapan utama, dimana tahapan pertama adalah proses awal, tahapan kedua adalah ekstraksi fitur, dan tahapan ketiga adalah klasifikasi. Arsitektur sistem pengenalan bahasa isyarat SIBI secara umum dapat dilihat pada Gambar 4.

A. Proses awal

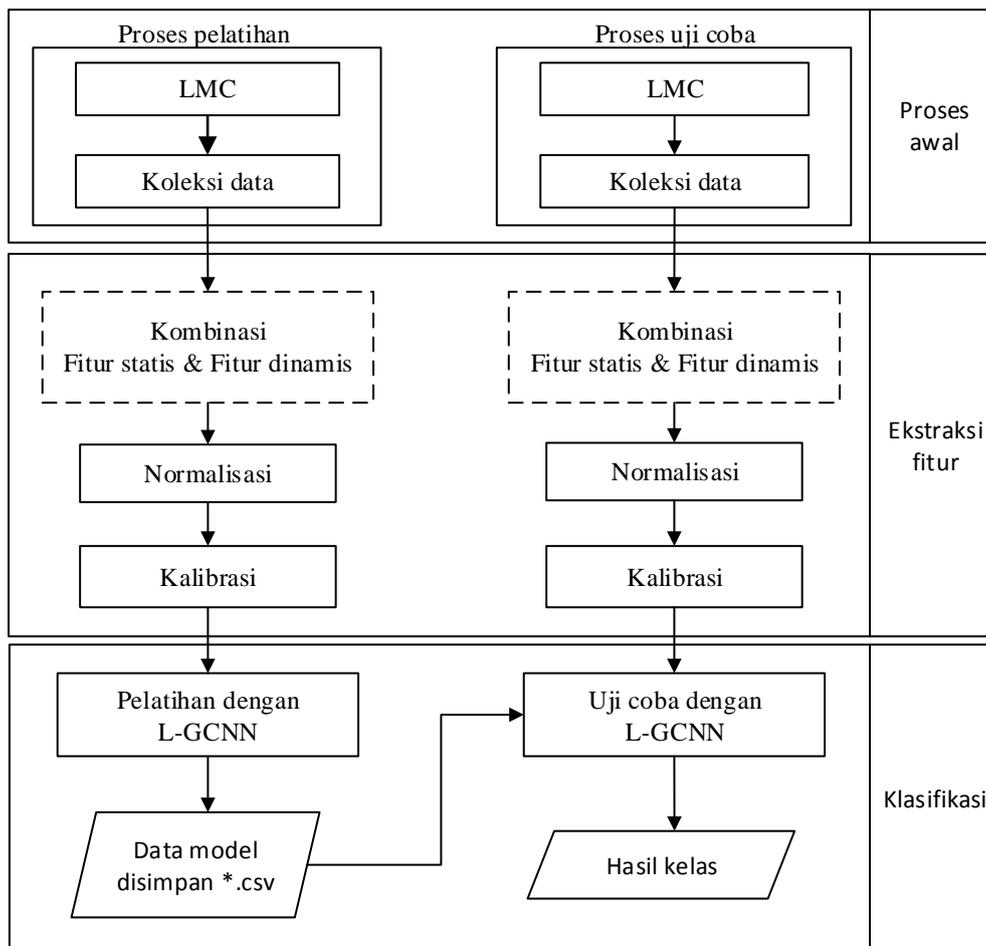
Proses awal yang dilakukan adalah untuk mendapatkan titik-titik koordinat tangan dengan menggunakan LMC. Titik-titik koordinat tangan yang digunakan pada penelitian sebanyak 10 frame. Penggunaan 10 *frame* karena fitur dinamis membutuhkan jumlah frame yang sesuai untuk membentuk gerakan dari sebuah huruf. Untuk membentuk huruf *J* membutuhkan gerakan yang pendek dan huruf *Z* membutuhkan gerakan yang lebih panjang dari pada huruf *J* sehingga penulis menentukan jumlah frame yang tepat adalah 10 *frame*.

B. Ekstraksi fitur

Ada dua jenis fitur yang digunakan dalam penelitian ini yaitu fitur yang bersifat statis dengan fitur yang bersifat dinamis. Fitur yang bersifat statis dibentuk berdasarkan hasil dari titik-titik koordinat tangan dalam posisi diam atau tidak bergerak. Sedangkan fitur yang bersifat dinamis adalah fitur yang dibentuk berdasarkan hasil dari titik-titik koordinat tangan dalam posisi bergerak.

Fitur statis yang digunakan pada penelitian ini adalah fitur *average spread*, *average tri-spread* dan *extended distance*. Fitur statis digunakan untuk mengenal bahasa isyarat yang bersifat statis. Adapun perhitungan dari fitur-fitur tersebut adalah:

- 1) *Average spread* adalah jarak rata-rata antara dua tip yang berdekatan. Perhitungan *average spread* dapat dilihat pada Persamaan (2).
- 2) *Average tri-spread* adalah luas rata-rata segitiga dari dua jari yang berdekatan. Perhitungan untuk *average tri-spread* dapat dilihat pada Persamaan (3).
- 3) *Extended distance* adalah jarak maksimal dari semua tulang jari seperti metacarpal, intermediate, dan distal terhadap palm center (titik tengah tangan).



Gambar 4. Arsitektur sistem pengenalan SIBI.

Fitur dinamis yang digunakan pada penelitian ini adalah *hand dynamic gesture*. Fitur dinamis digunakan untuk mengenal bahasa isyarat yang bersifat dinamis. Adapun perhitungan dari *hand dynamic gesture* dapat dilihat pada Persamaan (5), (6), dan (7).

C. Normalisasi

Normalisasi data adalah proses transformasi data ke-range [0-1] dengan tujuan agar data lebih proporsional. Normalisasi data fitur dihitung dengan menggunakan Persamaan (8).

D. Klasifikasi

Untuk proses klasifikasi sebagai pengenalan bahasa isyarat SIBI pada penelitian ini digunakan metode L-GCNN. Ada dua tahap proses klasifikasi pada L-GCNN yaitu proses pelatihan data dan proses uji coba data:

- 1) Pelatihan adalah proses untuk melatih dataset. Adapun proses pelatihan data dapat ditunjukkan pada Algoritma 1.
- 2) Uji coba adalah proses untuk menguji coba data terhadap data model yang dihasilkan dari proses pelatihan. Adapun proses uji coba dapat dilihat pada Algoritma 2.

IV. HASIL UJI COBA DAN ANALISA

A. Lingkungan Uji Coba

Sebagai 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 mempengaruhi sensor pada LMC dalam proses perekaman titik-titik koordinat pada tangan. *Leap motion visualizer* adalah API dari *developer* yang memvisualisasikan 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*.

B. 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.

C. Skenario Uji Coba

Skenario uji coba pada penelitian ini dibagi kedalam dua sub skenario yaitu skenario A dan skenario B sebagai berikut:

- 1) Skenario A adalah skenario uji coba *real time* yang dilakukan dengan hanya menggunakan fitur statis yang didapatkan dari LMC dan metode L-GCNN.
- 2) Skenario B adalah skenario uji coba *real time* yang dilakukan dengan menggunakan kombinasi fitur statis dan fitur dinamis yang didapatkan dari LMC dan metode 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 (23).

$$Akurasi = \frac{N_{cor}}{N_{tes}} \times 100\%, \quad (23)$$

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

D. Hasil Uji Coba

Untuk skenario uji coba A, 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 A dapat dilihat pada Gambar 5.

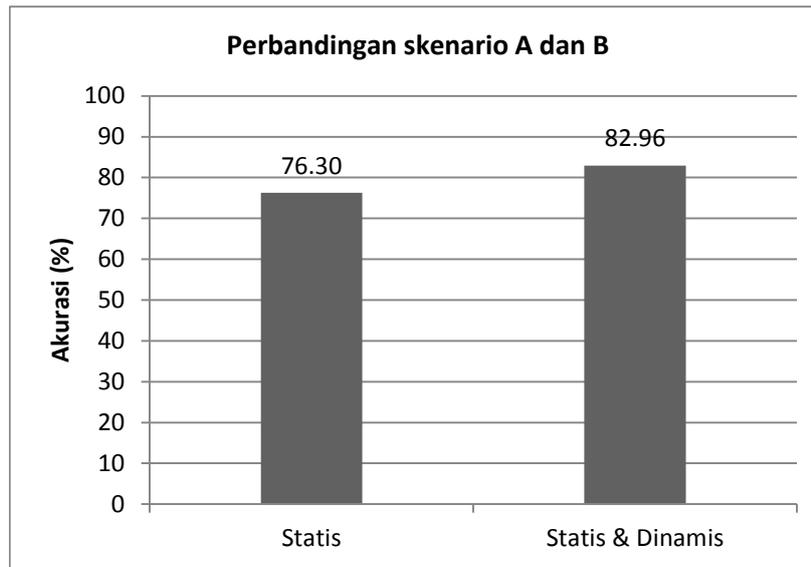
Pada Gambar 5 dapat dilihat bahwa akurasi pengenalan cukup baik untuk beberapa huruf seperti huruf NR, B, C, F, G, L, O, Q, S, T, U, V dan X karena huruf tersebut memiliki fitur yang tingkat kemiripannya cukup rendah dibandingkan dengan huruf-huruf yang lain. Beberapa kali terjadi kesalahan pengenalan 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 B, pengujian dilakukan secara *real time* dengan menggunakan kombinasi fitur statis dan fitur dinamis dengan metode L-GCNN. Pengujian untuk setiap huruf dilakukan sebanyak sepuluh kali dengan menggunakan tangan penulis. Hasil uji coba skenario B dapat dilihat pada **Error! Reference source not found.**

Pada **Error! Reference source not found.** 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.

Hasil skenario uji coba A dan B digambarkan pada grafik seperti pada Gambar 7. Pada Gambar 7 dapat dilihat bahwa dapat dibandingkan pengenalan bahasa isyarat dengan menggunakan fitur statis dan menggunakan kombinasi 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.



Gambar 7. Hasil perbandingan skenario A dan skenario B.

E. Pembahasan

Dari hasil pengujian yang dilakukan pada skenario A dan skenario B dapat disimpulkan bahwa pengenalan bahasa isyarat dengan menggunakan kombinasi fitur statis dengan fitur dinamis memiliki peningkatan akurasi pengenalan hingga 6,67%. Hal ini terjadi karena pengenalan bahasa isyarat dengan hanya menggunakan fitur statis hanya dapat mengenal bahasa isyarat yang bersifat statis dan tidak dapat mengenal bahasa isyarat yang bersifat dinamis dengan baik. Padahal bahasa isyarat SIBI memiliki dua sifat yaitu bahasa isyarat yang bersifat statis dan bahasa isyarat yang bersifat dinamis.

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. Peningkatan akurasi pengenalan ketika menggunakan kombinasi fitur statis dengan fitur dinamis hanya 6,67% karena pada huruf A sampai Z hanya terdapat dua huruf yang bersifat dinamis yaitu J dan Z. Ada beberapa huruf statis yang tidak dapat dikenal dengan baik, karena beberapa huruf tersebut memiliki tingkat kemiripan yang tinggi. Adapun huruf yang memiliki kemiripan yang tinggi pertama adalah huruf A, M, N, O, dan S. Untuk huruf yang memiliki tingkat kemiripan yang tinggi kedua adalah I dan Y. Huruf-huruf yang memiliki tingkat kemiripan yang tinggi 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 selalu tertutup oleh jari lainnya.

V. KESIMPULAN DAN SARAN

A. 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) Kombinasi fitur statis dengan fitur dinamis dapat meningkatkan akurasi pengenalan bahasa isyarat hingga 6.67%.

B. 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 tinggi seperti huruf (A, E, M, N, S) dan (K dengan V).
- 2) Perlu adanya penelitian lebih lanjut tentang kalibrasi dari jari tangan setiap pengguna karena ukuran tangan setiap pengguna berbeda sehingga dapat mempengaruhi hasil fitur.

DAFTAR PUSTAKA

- [1] M. M. Zaki and S. I. Shaheen, "Sign language recognition using a combination of new vision based features," *Pattern Recognit. Lett.*, vol. 32, no. 4, pp. 572–577, 2011.
- [2] R. P. Sharma and G. K. Verma, "Human Computer Interaction using Hand Gesture," *Procedia - Procedia Comput. Sci.*, vol. 54, pp. 721–727, 2015.
- [3] Y. Zhou, G. Jiang, and Y. Lin, "A novel finger and hand pose estimation technique for real-time hand gesture recognition," *Pattern Recognit.*, vol. 49, pp. 102–114, 2016.
- [4] A. Z. Shukor, M. F. Miskon, M. H. Jamaluddin, F. bin Ali@Ibrahim, M. F. Asyraf, and M. B. bin Bahar, "A New Data Glove Approach for Malaysian Sign Language Detection," *Procedia Comput. Sci.*, vol. 76, no. Iris, pp. 60–67, 2015.
- [5] C. Oz and M. C. Leu, "Linguistic properties based on American Sign Language isolated word recognition with artificial neural networks using a sensory glove and motion tracker," *Neurocomputing*, vol. 70, no. 7, pp. 2891–2901, 2007.
- [6] Y. Chen, Z. Ding, Y. Chen, and X. Wu, "Rapid Recognition of Dynamic Hand Gestures using Leap Motion," no. August, pp. 1419–1424, 2015.
- [7] C. Chuan, E. Regina, C. Guardino, A. L. M. Controller, and I. Apis, "American Sign Language Recognition Using Leap Motion Sensor," pp. 541–544, 2014.
- [8] B. Melis and M. Avci, "Generalized classifier neural network," *Neural Networks*, vol. 39, pp. 18–26, 2013.
- [9] B. Melis and M. Avci, "Logarithmic learning for generalized classifier neural network," *Neural Networks*, vol. 60, pp. 133–140, 2014.
- [10] G. Marin, F. Dominio, and P. Zanuttigh, "Hand gesture recognition with leap motion and kinect devices," *2014 IEEE Int. Conf. Image Process.*, pp. 1565–1569, 2014.
- [11] a. S. Elons, M. Abull-Ela, and M. F. Tolba, "A proposed PCNN features quality optimization technique for pose-invariant 3D Arabic sign language recognition," *Appl. Soft Comput. J.*, vol. 13, no. 4, pp. 1646–1660, 2013.
- [12] M. Mohandes, S. Aliyu, and M. Deriche, "Arabic Sign Language Recognition using the Leap Motion Controller," 2014.
- [13] M. Mohandes, S. Aliyu, and M. Deriche, "Prototype Arabic Sign Language Recognition using Multi-Sensor Data Fusion of Two Leap Motion Controllers," pp. 1–6, 2015.
- [14] A. S. Elons, M. Ahmed, H. Shedid, and M. F. Tolba, "Arabic Sign Language Recognition Using Leap Motion Sensor," pp. 368–373, 2014.
- [15] N. Sugianto and F. Samopa, "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)," vol. 01, no. 01, pp. 56–72, 2015.
- [16] W. N. Khotimah, R. A. Saputra, N. Suciati, and R. R. Hariadi, "Comparison between Back Propagation Neural Network and Genetic Algorithm Back Propagation Neural Network for Sign Language Recognition," *The International Conference on Science and Information Technology*, Yogyakarta, 2015.