PENERAPAN *NEURAL NETWORK BACKPROPAGATION* UNTUK PENCOCOKAN TULISAN TANGAN HURUF HIJAIYAH PADA VISUALISASI *MAKHORIJUL HURUF* BERBASIS *AUGMENTED REALITY*

SKRIPSI

Oleh:
MOHAMMAD YUSUF HIDAYAT
NIM: 11650079

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

2015

PENERAPAN *NEURAL NETWORK BACKPROPAGATION* UNTUK PENCOCOKAN TULISAN TANGAN HURUF HIJAIYAH PADA VISUALISASI *MAKHORIJUL HURUF* BERBASIS *AUGMENTED REALITY*

SKRIPSI

Oleh:
MOHAMMAD YUSUF HIDAYAT
NIM: 11650079

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

20

HALAMAN PENGAJUAN

PENERAPAN *NEURAL NETWORK BACKPROPAGATION* UNTUK PENCOCOKAN TULISAN TANGAN HURUF HIJAIYAH PADA VISUALISASI *MAKHORIJUL HURUF* BERBASIS *AUGMENTED REALITY*

SKRIPSI

Diajukan kepada:

Fakultas Sains dan Teknologi

Universitas Islam Negeri Maulana Malik Ibrahim Malang

Untuk Memenuhi Salah Satu Persyaratan Dalam

Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh:

MOHAMMAD YUSUF HIDAYAT

NIM: 11650079

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2015

HALAMAN PERSETUJUAN

PENERAPAN *NEURAL NETWORK BACKPROPAGATION* UNTUK PENCOCOKAN TULISAN TANGAN HURUF HIJAIYAH PADA VISUALISASI *MAKHORIJUL HURUF* BERBASIS *AUGMENTED REALITY*

SKRIPSI

Oleh :

Nama : Mohammad Yusuf Hidayat
NIM : 11650079
Jurusan : Teknik Informatika
Fakultas : Sains Dan Teknologi

Telah Disetujui, 28 Oktober 2015

Dosen Pembimbing I	**Dosen Pembimbing II**
Fresy Nugroho, M.T	**Irwan Budi Santoso, M.Kom**
NIP. 19710722 201101 1 001	**NIP. 19770103 201101 1 004**

Mengetahui,

Ketua Jurusan Teknik Informatika

Dr. Cahyo Crysdian

NIP. 19740424 200901 1 008

HALAMAN PENGESAHAN

PENERAPAN *NEURAL NETWORK BACKPROPAGATION* UNTUK PENCOCOKAN TULISAN TANGAN HURUF HIJAIYAH PADA VISUALISASI *MAKHORIJUL HURUF* BERBASIS *AUGMENTED REALITY*

SKRIPSI

Oleh :
Mohammad Yusuf Hidayat
NIM. 11650079

Telah Dipertahankan Di Depan Dewan Penguji Skripsi
Dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)

Tanggal 5 November 2015

Susunan Dewan Penguji:		**Tanda Tangan**
1. Penguji Utama	**: Fachrul Kurniawan, M.MT** **NIP. 19771020 200901 1 001**	**( )**
2. Ketua Penguji	**: Dr. Cahyo Crysdian** **NIP. 19740424 200901 1 008**	**( )**
3. Sekretaris Penguji	**: Fresy Nugroho, M.T** **NIP. 19710722 201101 1 001**	**( )**
4. Anggota Penguji	**: Irwan Budi Santoso, M.Kom** **NIP. 19770103 201101 1 004**	**( )**

Mengetahui,

Ketua Jurusan Teknik Informatika

Dr. Cahyo Crysdian

NIP. 19740424 200901 1 008

HALAMAN PERNYATAAN

ORISINALITAS PENELITIAN

Saya yang bertandatangan di bawah ini:

Nama	: Mohammad Yusuf Hidayat
NIM	: 11650079
Fakultas/Jurusan	: Sains Dan Teknologi / Teknik Informatika
Judul Penelitian	: Penerapan *Neural Network Backpropagation* Untuk Pencocokan Tulisan Tangan Huruf Hijaiyah Pada Visualisasi Makhorijul Huruf Berbasis *Augmented Reality*

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan atau pikiran oarang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka. Apabila di kemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut

Malang, 28 Oktober 2015

Yang Membuat Pernyataan,

Mohammad Yusuf Hidayat

1165079

MOTTO

Dari Skripsi Aku Belajar

Terkadang Allah menurunkan suatu "ilmu" dan "hikmah"
melalui permasalahan dan hambatan yang dihadapi.

"KHUSNUDZON"

ITU MENENANGKAN HATI

HALAMAN PERSEMBAHAN

Dengan rasa syukur dan mengharap ridlo **Allah**

Kupersembahkan karya ini untuk :

Ayahanda dan Ibunda tercinta

Ainuddin dan **Rusiana**

Yang senantiasa mencurahkan kasih sayang, perhatian, al-fatihah, doa, dan bimbingannya disetiap langkahku

Semoga Allah SWT selalu melindungi dan menyayangi keduanya

Sahabat-sahabatku semua yang sudah rela kuganggu untuk membantu Perjuanganku

Luqman, Wildan, Dani. *Sepurane Bro Wes Gupui peyan-peyan.* Dan seluruh kawan-kawan **INTEGER'11**.

Terima kasih sobat.

Semoga Allah menyayangi kita semua

Aamiin..

KATA PENGANTAR

Segala puji bagi Allah SWT yang telah melimpahkan rahmat, hidayah serta inayah-Nya kepada penulis sehingga bisa menyelesaikan skripsi dengan judul "Penerapan *Neural Network Backpropagation* Untuk Pencocokan Tulisan Tangan Huruf Hijaiyah Pada Visualisasi *Makhorijul Huruf* Berbasis *Augmented Reality*".

Shalawat serta salam semoga tercurah kepada Nabi Agung Muhammad SAW yang telah membimbing umatnya dari gelapnya kekufuran menuju cahaya Islam yang terang benderang.

Penulis menyadari keterbatasan pengetahuan yang penulis miliki, oleh karena itu tanpa keterlibatan dan sumbangsih dari berbagai pihak, sulit bagi penulis untuk menyelesaikan skripsi ini. Maka dari itu dengan segenap kerendahan hati patutlah penulis ucapkan terima kasih kepada:

1. Bapak, Ibu, dan Kakak tercinta yang selalu memberi dukungan yang tak terhingga serta menberi do'a yang senantiasa mengiringi setiap langkah penulis
2. Fresy Nugroho, M.T, selaku dosen pembimbing I yang telah meluangkan waktu untuk membimbing, memotivasi, mengarahkan dan memberi masukan dalam pengerjaan skripsi ini.

3. Irwan Budi Santoso, M.T, selaku dosen pembimbing II, yang selalu memberikan masukan, nasehat serta petunjuk dalam penyusunan laporan skripsi ini.
4. Bapak Dr. Cahyo Crysdian, selaku Ketua Jurusan Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang.
5. Bapak Mohammad Amin, MT, selaku dosen wali
6. Segenap Dosen Teknik Informatika yang telah memberikan bimbingan keilmuan kepada penulis selama masa studi.
7. Keluarga besar Perpustakaan Pusat UIN Maulana Malik Ibrahim Malang, yang telah memberi kesempatan serta dukungan kepada penulis untuk menyelesaikan skripsi ini.
8. Seluruh civitas akademika UIN Maliki Malang, khususnya jurusan Teknik Informatika angkatan 2011 yang telah memberikan banyak pengalaman berharga bagi penulis.
9. Semua pihak yang tidak mungkin penulis sebutkan satu-persatu, atas segala yang telah diberikan kepada penulis dan dapat menjadi pelajaran.

Sebagai penutup, penulis menyadari dalam skripsi ini masih banyak kekurangan dan jauh dari sempurna. Semoga apa yang menjadi kekurangan bisa disempurnakan oleh peneliti selanjutnya. Apa yang menjadi harapan penulis, semoga karya ini bermanfaat bagi kita semua. Aamiin.

Malang, 28 Oktober 2015

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Hidayat, M. Yusuf. 2015. **Penerapan *Neural Network Backpropagation* Untuk Pencocokan Tulisan Tangan Huruf Hijaiyah Pada Visualisasi *Makhorijul Huruf* Berbasis *Augmented Reality*.** Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang, Pembimbing: : (I) Fresy Nugroho, M.T (II) Irwan Budi Santoso, M.Kom

---

Kata Kunci : *Huruf Hijaiyah, Neural Network Backpropagation, Augmented Reality, Makhorijul Huruf, Pengenalan Tulisan Tangan*

Pembelajaran cara pengucapan *makhorijul huruf* untuk huruf hijaiyah sebenarnya lebih efektif ketika langsung berhadapan dengan guru yang biasa disebut *tallaqi Huruf Hijaiyah, Neural Network Backpropagation, Augmented Reality, Makhorijul Huruf, Pengenalan Tulisan Tangan*atau *musyafahah*, namun beberapa masih membutuhkan penunjang untuk membantu kefektifan dalam prosesnya. Penggunaan teknologi *Augmented Reality* diharapkan menjadi salah satu alternatif ketika seorang murid ingin mengetahui cara pengucapan huruf dengan melihat animasi yang memperlihatkan organ dalam mulut. Pada penelitian ini dikembangkan *Augmented Reality* yang dibangun dengan *platform Android* berupa aplikasi kamera untuk membantu proses pengajaran ilmu tajwid khusus pada cara pengucapan huruf atau *makhorijul huruf.* Metode yang digunakan untuk membangun sistem ini adalah *Neural Network Backpropagation.* Metode ini digunakan untuk proses pengenalan tulisan tangan huruf hijayah. Hasil *training* menunjukkan bahwa objek tulisan tangan mampu dikenali setelah dilakukan iterasi sebanyak 3738 kali dengan *learning rate 0.7f, dan stop condition* mencapai 0.0000000000001. Penghentian iterasi menggunakan perbandingan antara MSE dan *learning rate* dan jumlah *max run* sejumlah 50000. Prosentase keberhasilan *testing* pencocokan huruf hijaiyah 67.5% pada device *Android* Zenfone 4.

ABSTRACT

Hidayat, M. Yusuf. 2015. ***Implementation of Neural Network Backpropagation For Handwritting of Hijaiyah Letter Recognition on Visualization of Makhorijul Huruf Based of Augmented Reality.*** Department of Information Enginering, Faculty of Science and Technology, State Islamic University of Maulana Malik Ibrahim Malang, Supervisor: (I) Fresy Nugroho, M.T (II) Irwan Budi Santoso, M.Kom

---

Keywords*: Hijaiyah letter, Neural Network Backpropagation, Augmented Reality, Makhorijul Huruf, Handwritting Recognition*

Learning pronunciation of makhorijul huruf for hijaiyah letter actually more effective when dealing directly with teachers commonly called *tallaqi of musyafahah,* but some process still requiring the support to aid effectiveness of the process. The use of Augmented Reality technology is expected to be an alternative way when a student wants to know how the pronunciation of the hijaiyah letter by seeing an animation that shows the inside organ of the mouth. In this case, developed the Augmented Reality that's built by Android Platform under the form of camera application to assist tajwid teaching process especially for pronunciation of makhorijul huruf. The method used to build this system is a Neural Network Backpropagation algorithm. This methid is used to recognize the handwriting of hijaiyah letters. Training results indicate that the object of handwritting is able to recognize after doing 3738 times iterations with learning rate 0.7f, and stop condition reached 0.0000000000001. Termination of iterations using a comparison between the MSE value and the learning rate and the max number of iteration 50000. The percentage of succesfull testing results reached 67.5% on Zenfone 4 Android device.

مستخلس البحث

هداية، محمد يوسف، ٢٠١٥، تطبيق نيورال نيتوورك بايك بارا فا كا تييون لمناسبة كتابة حرف الهجائية على بصر مخارج الحروف من خلال اوكمينتيد رييياليتي البحث الجامعي في قسم هندسة المعلوماتيّة كلّية العلوم والتكنولوجيّا جامعة مولانا مالك إبراهيم الإسلاميّة الحكوميّة بمالانقو, المشريف : الأستاذ فرسي نغراها الماجستير والأستاذ إروان بودي سانطوسا الماجستير.

الكلمات الأساسيّة : *حرف الهجائيّة, نيورال نيتوورك بايك بارا فا كا تييون ,اوكمينتيد رييياليتي, مخارج الحروف, تعرّف كتابة حرف*

تعليم نطق مخارج حروف الهجائيّة أكثر فعّاليّة عند تعامل مباشرة مع المعلمين يطلق عليه التّلاقي أو المشافهة، ولكنه يحتاج الوسائل لمساعدة فعّاليّة خطوته. إستخدام تكنولوجيا اوكمينتيد رييياليتي يرجى أن يكون إحدى من المبادل كما يريد التلاميذ أن يعرف تعليم نطق الحروف بصور المتحرّكة يظهر الجهاز في الفمّ. هذا البحث تطوّر اوكمينتيد رييياليتي باعتماد فالات فورم اندرويد بشكل تطبيق المصوّر للمساعدة في عملية تعليم علم التجويد خاص على كيفية نطق الحرف أو مخارج الحروف. إستخدام هذا الطريقة لإعتماد منظّمة نيورال نيتوورك بايك بارا فا كا تييون تعرّف كتابة حرف الهجائيّة. نتيجة التدريب تدلّ عن الكتابة تستطيع معروفا بعد التكرار ٣٧٣٨ مرّة بمعدل التعلّم ٠,٧f ووقف الحالة يبلغ ٠,٠٠٠٠٠٠٠٠٠٠٠٠١ يستعمل وقف التكرار المقارنة بين "*MSE*"، ومعدل التعلّم، و عدد "*ماكس رون* " عدد ٥٠٠٠٠ نسبة مئويّة نتيجة تدريب مناسبة حرف الهجائيّة ٦٧% على جهاز زينفون اندرويد ٤.

مستخلس البحث

هداية، محمد يوسف، ٢٠١٥، تطبيق نيورال نيتوورك بايك بارا فا كا تييون لمناسبة كتابة حرف الهجائية على بصر مخارج الحروف من خلال اوكمينتيد رييااليتي البحث الجامعي في قسم هندسة المعلوماتيّة كلّية العلوم والتكنولوجيّا جامعة مولانا مالك إبراهيم الإسلاميّة الحكوميّة بمالانقو, المشريف : الأستاذ فرسي نغراها الماجستير والأستاذ إروان بودي سانطوسا الماجستير.

الكلمات الأساسيّة : *حرف الهجائيّة, نيورال نيتوورك بايك بارا فا كا تييون ,اوكمينتيد رييااليتي, مخارج الحروف, تعرّف كتابة حرف*

تعليم نطق مخارج حروف الهجائيّة أكثر فعّاليّة عند تعامل مباشرة مع المعلمين يطلق عليه التّلاقي أو المشافهة، ولكنه يحتاج الوسائل لمساعدة فعّاليّة خطوته. إستخدام تكنولوجيا اوكمينتيد رييااليتي يرجى أن يكون إحدى من المبادل كما يريد التلاميذ أن يعرف تعليم نطق الحروف بصور المتحرّكة يظهر الجهاز في الفمّ. هذا البحث تطوّر اوكمينتيد رييااليتي باعتماد فالات فورم اندرويد بشكل تطبيق المصوّر للمساعدة في عملية تعليم علم التجويد خاص على كيفية نطق الحرف أو مخارج الحروف. إستخدام هذا الطريقة لإعتماد منظّمة نيورال نيتوورك بايك بارا فا كا تييون تعرّف كتابة حرف الهجائيّة. نتيجة التدريب تدلّ عن الكتابة تستطيع معروفا بعد التكرار ٣٧٣٨ مرّة بمعدل التعلّم ٠,٧f ووقف الحالة يبلغ ٠,٠٠٠٠٠٠٠٠٠٠٠٠١ يستعمل وقف التكرار المقارنة بين "*MSE*"، ومعدل التعلّم، و عدد " *ماكس رون* " عدد ٥٠٠٠٠ نسبة مئويّة نتيجة تدريب مناسبة حرف الهجائيّة ٦٧% على جهاز زينفون اندرويد ٤.

ABSTRACT

Hidayat, M. Yusuf. 2015. ***Implementation of Neural Network Backpropagation For Handwritting of Hijaiyah Letter Recognition on Visualization of Makhorijul Huruf Based of Augmented Reality.*** Department of Information Enginering, Faculty of Science and Technology, State Islamic University of Maulana Malik Ibrahim Malang, Supervisor: (I) Fresy Nugroho, M.T (II) Irwan Budi Santoso, M.Kom

---

Keywords*: Hijaiyah letter, Neural Network Backpropagation, Augmented Reality, Makhorijul Huruf, Handwritting Recognition*

Learning pronunciation of makhorijul huruf for hijaiyah letter actually more effective when dealing directly with teachers commonly called *tallaqi of musyafahah,* but some process still requiring the support to aid effectiveness of the process. The use of Augmented Reality technology is expected to be an alternative way when a student wants to know how the pronunciation of the hijaiyah letter by seeing an animation that shows the inside organ of the mouth. In this case, developed the Augmented Reality that's built by Android Platform under the form of camera application to assist tajwid teaching process especially for pronunciation of makhorijul huruf. The method used to build this system is a Neural Network Backpropagation algorithm. This methid is used to recognize the handwriting of hijaiyah letters. Training results indicate that the object of handwritting is able to recognize after doing 3738 times iterations with learning rate 0.7f, and stop condition reached 0.0000000000001. Termination of iterations using a comparison between the MSE value and the learning rate and the max number of iteration 50000. The percentage of succesfull testing results reached 67.5% on Zenfone 4 Android device.

ABSTRAK

Hidayat, M. Yusuf. 2015. **Penerapan *Neural Network Backpropagation* Untuk Pencocokan Tulisan Tangan Huruf Hijaiyah Pada Visualisasi *Makhorijul Huruf* Berbasis *Augmented Reality*.** Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang, Pembimbing: : (I) Fresy Nugroho, M.T (II) Irwan Budi Santoso, M.Kom

---

Kata Kunci : *Huruf Hijaiyah, Neural Network Backpropagation, Augmented Reality, Makhorijul Huruf, Pengenalan Tulisan Tangan*

Pembelajaran cara pengucapan *makhorijul huruf* untuk huruf hijaiyah sebenarnya lebih efektif ketika langsung berhadapan dengan guru yang biasa disebut *tallaqi Huruf Hijaiyah, Neural Network Backpropagation, Augmented Reality, Makhorijul Huruf, Pengenalan Tulisan Tangan*atau *musyafahah*, namun beberapa masih membutuhkan penunjang untuk membantu kefektifan dalam prosesnya. Penggunaan teknologi *Augmented Reality* diharapkan menjadi salah satu alternatif ketika seorang murid ingin mengetahui cara pengucapan huruf dengan melihat animasi yang memperlihatkan organ dalam mulut. Pada penelitian ini dikembangkan *Augmented Reality* yang dibangun dengan *platform Android* berupa aplikasi kamera untuk membantu proses pengajaran ilmu tajwid khusus pada cara pengucapan huruf atau *makhorijul huruf*. Metode yang digunakan untuk membangun sistem ini adalah *Neural Network Backpropagation.* Metode ini digunakan untuk proses pengenalan tulisan tangan huruf hijayah. Hasil *training* menunjukkan bahwa objek tulisan tangan mampu dikenali setelah dilakukan iterasi sebanyak 3738 kali dengan *learning rate 0.7f, dan stop condition* mencapai 0.0000000000001. Penghentian iterasi menggunakan perbandingan antara MSE dan *learning rate* dan jumlah *max run* sejumlah 50000. Prosentase keberhasilan *testing* pencocokan huruf hijaiyah 67.5% pada device *Android* Zenfone 4.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan metodologi pembelajaran Al-Quran saat ini sangat pesat. Dahulu, kita hanya mengenal metode pembelajaran *iqra'* sebagai bahan pengajaran untuk anak-anak madrasah, TPQ/TPA. Namun sekarang kita sudah mengenal beberapa metode seperti *qiro'ati, tartila, ummi*, serta ada juga yang khusus untuk pembelajaran cara baca untuk Al-Quran dengan *rosm ustmany* yakni *bayan lilmuslimin.* Beberapa bahkan sudah memanfaatkan teknologi ketika pembelajaran tajwid. Seperti dengan game, pembuatan aplikasi flash, ataupun pembelajaran tajwid dengan menggunakan presentasi layar LCD. *Trend* pemakaian teknologi informasi dalam pembelajaran sudah menjadi hal yang sangat umum pada saat ini. Bahkan pada hari Rabu, 15 Oktober 2014 Wakil Presiden Republik Indonesia, Boediono meluncurkan Kuliah Dalam Jaringan (Daring) Indonesia Terbuka Terpadu, yakni pembelajaran berbasis teknologi dan informasi di Kementrian Pendidikan dan Kebudayaan. (Setiawan, 2014)

Dalam sebuah penelitian berjudul "Kesulitan Belajar Santri Pada Pembelajaran Al-Qur'an di Musholla Nurur Rahman Trasak Larangan Pamekasan", karya Maisarah, mahasiswi STAI Pamekasan disebutkan bahwa kesulitan belajar santri pada pembelajaran Al-Quran, yaitu belum mampu mempraktekan bacaaan mad dengan benar, yaitu terkadang bacaan mad tidak dibaca panjang, dan seharusnya dibaca pendek malah dibaca panjang. Santri juga masih banyak melakukan kesalahan dalam membaca hukum bacaan, yang dibaca

dengung dan yang tidak dibaca dengung. Dari segi makhorijul huruf yang seharusnya dibaca "ث", akan tetapi oleh santri dibaca "س".(Maisarah, 2012)

Menurut data dari Tim Pemantau Gerakan Pembelajaran Al-Qur'an tahun 2004, menunjukkan bahwa untuk daerah Sulawesi Selatan, khususnya dikalangan peserta didik setingkat SMA, hanya 15,88% siswa yang fasih dalam membaca Al-Qur'an. Sedangkan untuk setingkat SMP, hanya 18,30% siswa yang fasih membaca Al-Qur'an dan setingkat SD, hanya terdapat 18% yang fasih membaca Al-Qur'an, sementara 5,76% diantaranya sama sekali belum bisa membaca Al-Qur'an.

Diantara bagian paling inti dari membaca Al-Quran adalah tajwid, membaca dengan lagu yang tidak mengeluarkannya dengan tajwid. Bagian utama tajwid adalah makhorijul huruf dan bagaimana keluarnya huruf. Kalau kita salah membaca Al-Quran, maka artinya akan berubah. Hukum mengamalkan ilmu tajwid pada masalah harakat huruf dan tempat keluar huruf maka hukumnya wajib. Salah padanya haram secara mutlak (Ustadz Aiman Abdullah, Lc).

Adapun dasar dari Al-Quran yang mewajibkan seseorang membaca Al-Quran dengan memahami tajwid adalah surat Al-Muzammil ayat 4 yang berbunyi:

أَوْ زِدْ عَلَيْهِ وَرَتِّلِ الْقُرْءَانَ تَرْتِيلًا ٤

"atau lebih dari seperdua itu. Dan bacalah Al Quran itu dengan perlahan-lahan."

Di dalam Tafsir Ibnu Katsir karya Dr. Abdullah Bin Muhammad Bin Abdurrahman Bin Ishaq Alu Syaikh yang diterjemahkan oleh M. Abdul Ghoffar

E.M. dan Abu Ihsan al-Atsari dijelaskan bahwa yang dimaksud dari ayat ini adalah bacalah al-Quran itu dengan perlahan, sebab hal itu akan membantu dalam memahami dan merenunginya. Dan di awal penafsiran telah disampaikan beberapa hadits yang menunjukkan disunnahkannya bacaan tartil dan pengindahan suara ketika membaca al-Quran. Di dalam terjemahan lain yang diterjemahkan oleh H. Salim Bahreisy dan H. Said Bahreisy dikatakan ada beberapa sahabat yang ditanya tentang bacaan al-Quran Rosulullah. *Pertama*, Anas bin Malik r.a. ketika ditanya tentang bacaan Nabi Saw, ia menjawab bahwa Rasulullah Saw, jika membaca dengan tartil, memanjangkan *mad* "Bismillah, Arrahman dan Arrahiim". *Kedua*, Ummu Salamah r.a juga ditanya tentang bacaan Rasulullah Saw. Ia menjawab bahwa Rasulullah Saw, jika membaca tiap ayat berhenti. Rasulullah Saw, juga menganjurkan bila membaca Al-Quran hendaklah memerdukan suaranya : "*Zayyimul qurana bi ash waatikum*". Hiasilah Al-Quran dengan suaramu. "*Laisa minnaa man lam yataghanna bil qurani*". Bukan dari golonganku siapa yang tidak memerdukan suaranya dalam membaca Al-Quran. Selanjutnya Ibnu Masud r.a, juga berkata, " Janganlah membaca Al-Quran bagai menabur pasir, dan janganlah kamu baca bagai sajak syair, perhatikanlah isinya, bangkitkan perasaan, hati dan pikiran orang yang mendengarkan bacaaanmu itu, dan janganlah terburu berhenti pada akhir surat.

Di dalam Tafsir Quran Karim, karya Prof. Dr. H. Mahmud Yunus, kata (**ترتيلا**) diartikan dengan perlahan-lahan (terang huruf-hurufnya). Diterangkan pula di dalam buku Al-Quran dan Tafsirnya yang dikeluarkan oleh Kementrian Agama RI bahwa dalam ayat ini, Allah memerintahkan Nabi Muhammad supaya membaca

Al-Quran dengan seksama (*tartil*). Maksudnya ialah membaca Al-Quran dengan pelan-pelan, bacaan yang fasih, dan merasakan arti dan maksud dari ayat-ayat yang dibaca itu, sehingga berkesan di hati. Di dalam buku ini juga disebutkan bahwa 'Asiyah r.a. meriwayatkan bahwa Rasulullah SAW membaca Al-Quran dengan tartil, sehingga surah yang dibacanya menjadi lebih lama dari ia membaca biasa. Dalam hubungan ayat ini, al-Bukhari dan Muslim meriwayatkan dari Abdullah bin Mughaffal, bahwa ia berkata :

رِايت رسول الله صلي الله عليه سلم يوم فتح مكة علي ناقته يقرأ سورة الفتح فرجع في قراءته

(*رواه البخاري و مسلم عن عبد الله بن مغفل*)

"*Aku melihat Rasulullah SAW pada hari penaklukan kota Makkah, sedang menunggang unta beliau membaca Surah al-Fath dimana dalam bacaan itu melakukan tarji' (bacaan lambat dengan mengulang-ulang).* (Riwayat al-Bukhari dan Muslim dari Abdillah bin Mughaffal)

Beberapa dalil di atas menunjukkan bahwa ketika seorang muslim membaca Al-Quran maka harus diperhatikan cara bacanya dan kefasihannya agar arti yang dimaksud tidak melenceng dan ketika bacaannya fasih dan bagus, maka pembaca dan pendengar bisa merasakan bagaimana indahnya Al-Quran. Namun ada juga yang berpendapat lain. Di dalam buku ini juga dijelaskan bahwa pengarang buku *Fathul Bayan* berkata, " Yang dimaksud dengan tartil ialah kehadiran hati ketika membaca, bukan asal mengeluarkan bunyi dari tenggorokan dengan memoncong-moncongkan muka dan mulut dengan alunan lagu, sebagaimana kebiasaan yang

dilakukan pembaca-pembaca Al-Quran zaman sekarang. Membaca yang seperti itu adalah suatu bacaaan yang dilakukan orang-orang yang tidak mengerti agama.

Pengertian di atas sama-sama benar. Maka dari itu bisa ditarik kesimpulan bahwa Al-Quran seharusnya dibaca bukan hanya dengan *fasih*, indah, dan benar, akan tetapi juga harus menghadirkan hati ketika membaca. Membaca dengan suara yang indah dan fasih juga bisa membuat hati pembaca dan pendengarnya bergetar dan menghayati isi kandungan Al-Quran. Beberapa aplikasi dan *game* sudah dibuat untuk membantu proses pengajaran tajwid. Namun yang menjadi kesulitan adalah ketika pembelajaran *makhorijul huruf*, ada huruf yang cara mengeluarkannya mengikutkan organ bagian dalam mulut. Dari sini dapat disimpulkan bahwa untuk membantu mengajarkan Al-Quran khususnya pada pengajaran *makharijul huruf* perlu adanya aplikasi yang bisa memvisualisasikan bagaimana seharusnya posisi gerakan mulut ketika huruf keluar.

Augmented Reality atau realitas tertambah adalah salah satu topik populer yang di dunia teknologi informasi saat ini dan disebut–sebut sebagai teknologi terbaru dan tertinggi dari penggunaan *webcam*. *Augmented Reality* atau yang sering disebut dengan AR adalah suatu teknologi yang "menggabungkan" antara dunia maya (*virtual*) dan dunia nyata (*real*) dengan menggunakan bantuan perangkat digital secara *real-time*. Biasanya AR digunakan untuk memproyeksikan objek 3 dimensi atau 2 dimensi yang bersifat maya ke lingkungan nyata. Sekarang AR sudah sering digunakan di berbagai belahan dunia untuk berbagai kepentingan dan sudah dikembangkan untuk beberapa *platform* seperti *handphone* dan perangkat *mobile* yang lain, sehingga tidak terbatas untuk

platform komputer saja. Bahkan sekarang sudah banyak *operating system* dan perangkat lunak yang telah mendukung teknologi AR ini (Andriyadi, 2011). Oleh karena itu, dalam penelitian ini dilakukan pemanfaatan *Augmented Reality* dalam memvisualisasikan *makhorijul huruf* yang diharapkan bisa lebih menarik untuk pembelajaran ilmu membaca Al-Quran.

Handwritting recognition adalah salah satu dari sistem pengenalan yang dijabarkan oleh Konstantinos Koutrambas dan Sergios Theorodis, dalam bukunya *Pattern Recognition* disebutkan bahwa system berusaha untuk mengenali tulisan tangan manusia agar dapat ditanggapi dengan baik oleh *computer*. Dengan sistem ini, *image* berupa tulisan tangan seorang guru berupa huruf hijaiyah dikenali untuk kemudian dimunculkan object 2D yang mengucapkan *makhorijul huruf* dari huruf *makhorijul huruf* dari tulisan tangan yang dikenali. Algoritma yang dipakai adalah *Neural Network Backpropagation* (Koutrambas dan Theorodis, 2008). *Backpropagation* melatih jaringan untuk mendapatkan keseimbangan antara kemampuan jaringan untuk mengenali pola yang digunakan selama pelatihan serta kemampuan jaringan untuk memberikan respon yang benar terhadap pola masukkan yang serupa (tapi tidak sama) dengan pola yang dipakai selama latihan.(Siang, 2005). Dengan algoritma ini diharapkan pengenalan pada pola tulisan tangan akan semakin akurat meskipun pola yang diinputkan tidak terlalu mirip.

Melalui teknologi yang dimanfaatkan untuk visualisasi *makhorijul huruf* ini diharapkan membuat santri atau orang tertarik untuk bisa mempraktekan bagaiaman cara mengeluarkan huruf selain menggunakan metode *talaqqi* dan *musyafahah*.

1.2 Rumusan Masalah

Berdasarkan penjelasan latar belakang di atas maka perumusan masalah dalam penelitian ini adalah sebagai berikut :

a. Apakah algoritma *Neural Network Backpropagation* sesuai dengan penelitian tentang pengenalan gambar?
b. Bagaimana kestabilan proses sistem dalam mengenali gambar huruf hijaiyah?
c. Berapa akurasi dari pengujian pengenalan tulisan tangan huruf hijaiyah?

1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah sebagai berikut:

a. *Image* Huruf hijaiyah adalah marker untuk perbandingan kemiripan huruf yang ditulis langsung.
b. Huruf hijaiyah yang digunakan berjumlah 10 huruf yaitu *alif* sampai *ro*.
c. *Neural Network Backpropagation* dengan *1 hidden layer* digunakan sebagai algoritma pencocokan kemiripan huruf hijaiyah.

1.4 Tujuan Penelitian

Tujuan dalam penelitian ini adalah sebagai berikut :

a. Untuk mengetahui kesesuaian Algoritma *Nerual Network Backpropagation* untuk proses pengenalan tulisan tangan huruf hijaiyah.
b. Untuk mengetahui kestabilan dalam pengenalan *image* tulisan tangan huruf hijaiyah.
c. Untuk mengukur akurasi dari algoritma yang dipakai dalam pengenalan tulisan tangan huruf.

1.5 Manfaat Penelitian

Manfaat yang nantinya diharapkan dari pengembangan aplikasi ini adalah sebagai berikut:

a. Bagi penulis
    1. Menerapkan ilmu yang telah didapatkan di Teknik Informatika UIN Maliki Malang.
    2. Mendapatkan pemahaman lebih lanjut tentang pengembangan aplikasi menggunakan teknologi Augmented Reality dengan metode *Neural Network Backpropagation* dalam pencocokan *image*.
    3. Bisa sedikit berkontribusi dalam pendidikan Islam khususnya pembelajaran Al-Quran.
b. Bagi pengguna
    1. Mempelajari dan Mempraktekan secara langsung cara baca suatu huruf hijaiyah, ketika kesulitan mencari guru untuk *talaqqi* dan *musyafahah*.
    2. Sebagai bahan ajar yang menarik untuk pembelajaran *makhorijul huruf* di TPQ/TPA, madrasah, bahkan pesantren.

BAB II

KAJIAN PUSTAKA

1.1 Penelitian Terkait

Pada penelitian yang dilakukan oleh Hermantono dan Rachman Wahyu Purnawan yang berjudul "*Prediksi Produksi Kelapa Sawit Berdasarkan Kualitas Lalahn Menggunakan Model Artificial Neural Network (ANN)*" dijelaskan bahwa Model terbaik untuk ANN adalah 7-3-1, dengan iterasi 30000, laju pembelajaran = 0,9, momentum = 0.9, dan konstanta *gain*-nya = 0.9. Dalam pelatihan dan pengujian, sistem ANN mempunyai kelebihan yakni proses yang akurat, cepat, serta dapat meminimalisi kesalahan (Hermanto dan Purnawan, 2009).

Penelitian selanjutnya yakni tentang penggunaan *Neural Network Backpropagation* untuk pengenalan pola karakter huruf jawa. Penelitian yang dilakukan oleh Nazla, et al. Dijelaskan bahwa masing-masing sampel memiliki karakteristik *BPNN* yang berbeda-beda untuk mendapatkan hasil pelatihan yang terbaik. Rata-rata keakuratan *BPNN* dalam pengenalan pola karakter jawa adalah sebesar 99,563% untuk data sampel berupa data pelatihan, 61, 359% untuk data sampel di luar pelatihan, 75% untuk data sample data sampel dari data pelatihan dan di luar pelatihan (Nazla, et al., 2010).

Penelitihan selanjutnya tentang pengenalan karakter Arab menggunakan Neural Network yang dilakukan oleh Basem Alijla dan Kathrein Kwaik. Dijelaskan bahwa bahasa arab mempunyai perbedaan pada setiap daerah untuk style penulisannya yang pasti menambah kompleksitas dari proses pengenalannya. NN telah dibuktikan sebagai sebuah konsep yang kuat, dan dipertimbangkan sebagai

metode yang paling sukses untuk pengenalan tulisan tangan, khususnya karakter Arab. Mereka mengenalkan system OIAHC sebagai sebuah step yang menunjukkan pendekatan neural network untuk menyelesaikan masalah pengenalan tulisan tangan Arab (Alijla dan Kwaik, 2012).

Penelitihan berikutnya tentang pengenalan karakter tulisan tangan latin dengan Nerual Network Backpropagation yang dilakukan oleh Dompak Petrus Sinambela dan Sampe Hotlan Sitorus. Dari penelitian tersebut disimpulkan bahwa hasil pengujian pengenalan tulisan tangan untuk 5 orang, tingkat keakurasian program yaitu sebesar 87.2%. Dijelaskan bahwa pemilihan *learning rate*-nya sangat mempengaruhi pergeseran nilai pada bobot-bobot saat proses pelatihan jaringan syaraf tiruan. Sedangkan data bobot (*w*) dan bias ke *output layer* yang baru, mengalami pergeseran nilai yang cukup besar. Mereka memberi saran bahwa untuk meningkatkan akurasinya maka perlu ditambahkan jumlah karakter tulisan dan jumlah epoh untuk pelatihan *JST* minimal 100 orang (Sinambela dan Sitorus, 2013).

Pada *Augmented Relity*-nya, ada penelitian tentang games yang dilakukan oleh Balint, Z., Kiss, B., Magyari, B., & Simon. Penelitian ini membahas tentang perburuan atau pencarian (*Scavenger hunt & Treasure Hunt*). Dijelaskan bahwa mereka menggunakan framework *Augemented Relity* untuk pengembangannya, dan framework ini menyediakan banyak solusi menarik, dan juga bisa menampilkan transformasi pemandangan. Untuk permainan *treasure* misalnya, pengenalan citra digunakan untuk penentuan suatu lokasi apakah sudah tepat atau belum (Balint et al., 2012).

Selanjutnya, adalah penelitian AR yang dilakukan oleh Permana, Kuswardayan, & Hariadi yang diterapkan dalam game AR Game Tower Defense. Game yang dibangun disini merupakan pengembangan dari dua aplikasi AR sebelumnya yakni AR Defense (*Desktop Only*) dan AR Defender (*iOS only*). Pengembangan ini dibuat untuk platform Android dengan menggunakan *Vuforia Engine* dan memberi kecerdasan buatan untuk NPC yang ada, serta menggunakan penanda bebas agar memberikan keleluasaan pada pengembang untuk memilih gambar sebagai *marker* (Perman, et al., 2013).

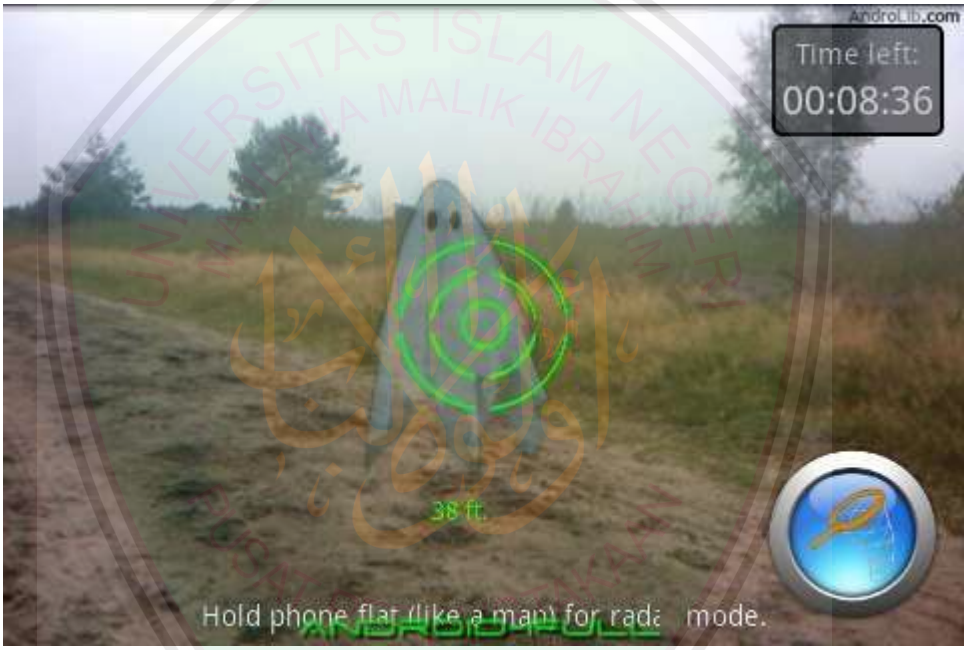
1.2 Augmented Reality

Augmented Reality, biasanya disebut dengan AR, telah menyimpan perhatian yang besar beberapa tahun ini. Istilah ini telah digunakan untuk mendiskripsikan teknologi dibalik perluasan atau intensifikasi dari dunia nyata. *"Augment reality"* adalah untuk mengintensifikasi atau memperluas realita atau kenyataan itu sendiri (Kreveren dan Poelman, 2010). Lebih spesifisik lagi, AR disebutkan sebagai suatu kemampuan untuk melapiskan media digital di atas dunia nyata melewati sebuah *screen* dari sebuah perangkat seperti PC (*Personal Computer*) atau sebuah *Smartphone*, untuk membuat dan menunjukkan kepada kita sebuah dunia yang penuh dengan informasi yang mungkin belum terkonsep sampai sekarang.

Di dalam buku "*Pro Augmented Reality"* karya Raghav Sood disebutkan bahwa ada beberapa kategori implementasi *Augmented Reality* (Sood, 2012):

a. *Casual Userne*

Beberapa aplikasi berbasis *Augmented Reality* digunakan oleh kebanyakan orang, misalnya *game*, peta, aplikasi navigasi. Aplikasi tersebut biasanya menggunakan *accelerometer,* dan GPS untuk mendapatkan lokasi dan keberadaan sesungguhnya dari *device*. Salah satunya adalah game *SpecTrek.* Game ini menggunakan *GPS* user untuk menemukan lokasi *user* dan mempelihatkan hantu yang kemudian akan *user* buru di sekitar area *user*.



Gambar 2. 1 Game SpecTrek (Sood, 2012)

b. *Military and Law Enforcement*

Di dalam kemiliteran dan pelaksanaan hukum menggunakan *AR goggles* untuk simulasi *training*. Mereka mempunyai simulator yang diterapkan dalam layar yang besar atau sebuah kendaraan yang diberi implementasi sebuah AR teknologi. Di dalam simulator itu beberapa scenario atau misi ditunjukkan, dan harus diselesaikan sebagai bentuk latihan.

c. *Vehicle*

Pada kendaraan, teknologi AR biasanya diterapkan pada multiplescreen yang masing-masing memperlihatkan *direction.* Jika pada kendaraan cuma ada satu layar dan banyak kamera, maka kendaraan akan memilih secarah otomatis, atau mengikuti perintah user. Gambar- gambar pada layar ditampilkan dengan data yang berguna seperti *small map, compass, direction arrows, alternate routes, weather forcast,* dan masih banyak lagi. Teknologi-teknologi seperti ini masih diterapkan pada pesawat terbang dan kereta. *Smart Cars* atau mobil pintar masih diuji coba untuk pasar.

d. *Medical*

Augmented Reality untuk pembedahan akan menjadi biasa pada masa kini. Pembedahan akan selesai dengan kesalahan paling kecil karean computer telah menghitung inputan yang sesuai untuk menggerakan robot untuk melakukan semua pembedahan. AR machine juga bisa untuk memonitor banyaknya pasien dan meyakinkan bahwa tanda-tanda bahaya pada pasien dibawah pengawasan setiap waktu.

e. *Trial Rooms*

Pada beberapa tempat perbelanjaan sudah dilengkapi dengan *virtual* ruangan. *User* atau pembeli akan berdiri di depan camera, kemudian user memilih baju yang akan dicoba. Maka computer akan meng-*augment* baju tersebut dan menampilkan pada layar.

f. *Education*

Pemanfaatan AR dalam dunia pendidikan biasanya dipakai dalam teks book yang di dalamnya terdapat material-material seperti bumi. Maka AR akan menampilkan objek tersebut dalam 3D untuk menunjukkan bentuk yang sebenarnya. Dan biasanya juga menggunakan animasi agar siswa bisa melihat proses yang sebenarnya sesuai urutan.

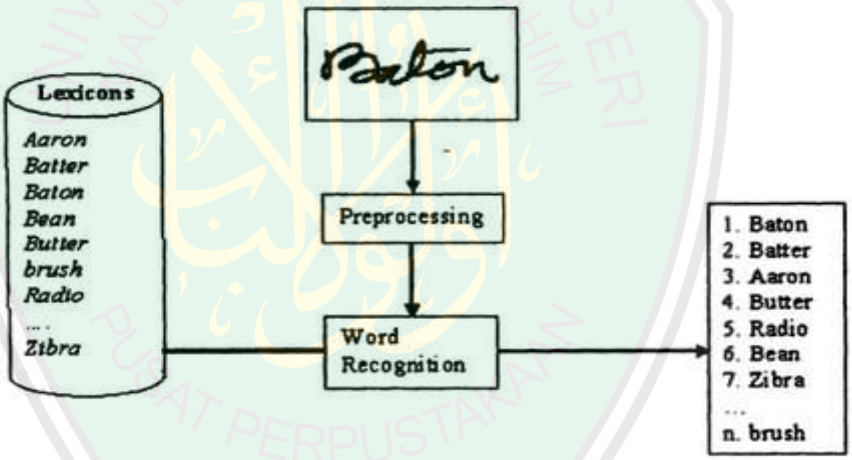
Gambar 2. 2 AR untuk edukasi

Dan masih banyak lagi pemanfaatan AR yang lain, misal pada *Tourism, Architecture, Cinema/Performance, Entertainment, Art, Weather Forcasting, Astronomy,* dan lain-lain.

1.3 Handwritting Recogintion

Computer Recognition atau pengenalan computer pada sebuah karakter atau kata adalah salah satu dari beberapa aplikasi yang sukses di *computer vision.* Secara sederhana, pengenalan ini adalah sebuah proses otomatis yang menggunakan pengenalan pola dan teknik *machine learning* untuk mengenali karakter atau kata

yang diberikan sebuah kamus. Dengan perkembangan kecepatan perhitungan, memory, teknologi *scanning device* yang tinggi, dan teknik pengenalan, dewasa imi kita telah melihat progress dari pengembangan *handwriting recognition.* Ada dua 2 domain masalah utama dalam pengenalan pola tulisan yakni: *on-line* dan *off-line*. Pada *on-line handwriting recognition*, proses dilakukan sejak input mulai diterima, karena teknik ini membutuhkan *timing information* / urutan waktu masuknya data. Sedangkan *off-line handwriting recognition*, proses berkerja setelah dilakukan setelah input sudah tersedia, maka dari itu informasi urutan waktu masuknya data tidak diketahui.(Liu, et al., 2003)



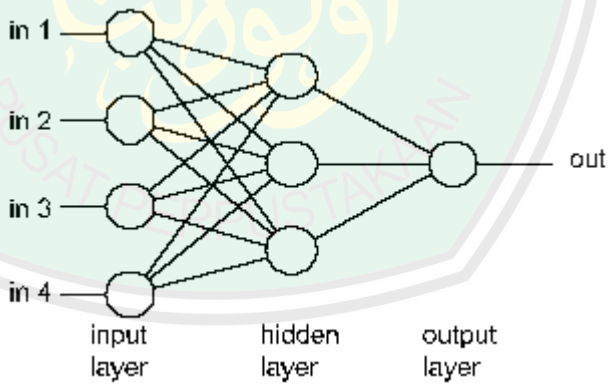
Gambar 2. 3 Handwriting word recognition system (Liu, et al, 2003)

1.4 Neural Network Backpropagation

Neural network atau biasa disebut jarisan syaraf tiruan (JST) sederhana pertama kali diperkenalkan oleh McCulloh dan Pitts di tahun 1943. McCulloh dan Pitts menyimpulkan bahwa kombinasi beberapa neuron sederhana menjadi sebuah system neural akan meningkatkan kemampuan komputasinya (Siang, 2005). JST didefinisikan sebagai suatu system pemrosesan informasi yang mempunyai

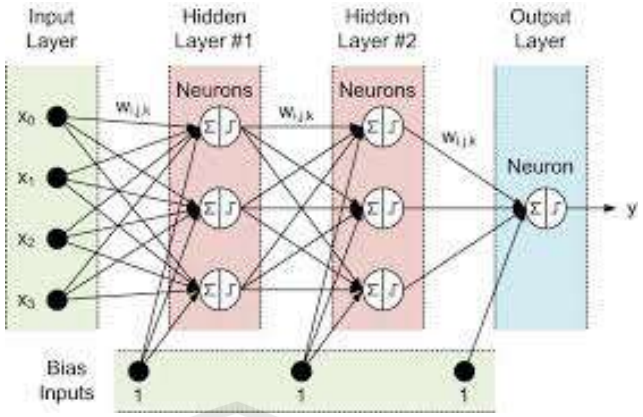
karakteristik menyerupai jaringan syaraf manusia. Dengan kata lain operasi dari algoritma ini diilhami dari pengetahuan tentang sel saraf biologis di dalam otak, yang merupakan salah satu representasi buatan dari otak manusia yang selalu mencoba menstimulasi proses pembelajaran pada otak manusia (Hermawan, 2006).

Pelatihan backpropagation termasuk dalam pelatihan terbimbing dan di desain untuk JST *feedforward* lapis jamak (*multi-layer*). Algoritma ini banyak dipakai pada aplikasi pengendalian karena proses pelatihannya didasarkan pada interkoneksi yang sederhana, yaitu: Jika keluaran memberikan hasil yang salah, maka penimbang (*weight*) dikoreksi supaya galatnya dapat diperkecil dan tanggapan JST selanjutnya akan lebih mendekati nilai yang benar. BP juga berkemampuan untuk memperbaiki penimbang pada lapis tersembunyi (*hidden layer*) (Purnomo dan Kurniawan, 2006).



Gambar 2. 4 Neural network backpropagation 1 layer hidden

(www.cheshireeng.com, 2003)



Gambar 2. 5 Neural network backpropagation 2 layer hidden

(www.cheshireeng.com, 2003)

Secara garis besar, algoritma ini disebut sebagai propagasi balik karena dapat diuraikan sebagai berikut: Ketika JST diberikan pola masukkan sebagai pola pelatihan maka pola tersebut menuju ke unit-unit pada lapis tersembunyi untuk diteruskan ke unit-unit lapis keluaran. Kemudian unit-unit lapis keluaran memberikan tanggapan yang disebut sebagai keluaran JST. Saat keluaran JST tidak sama dengan keluaran yang diharapkan maka keluaran akan disebarkan mundur (*backward*) pada lapis tersembunyi diteruskan ke unit pada lapis masukan (Purnomo dan Kurniawan, 2006).

Penambahan bias ini mengatasi jika pola inputan yang masuk adalah 0. Menurut (Fröhlich, 2015), jika semua nilai dari suatu pola inputan adalah 0, maka bobot V tidak akan pernah berubah untuk pola ini dan jaringan tidak bisa mempelajarinya.

Untuk lebih jelasnya rincian dari algoritma JST *Backpropagation* adalah sebagai berikut:

- ***Algoritma pelatihan Backpropagation***

  - **Langkah 0**

    Pemberian inisialisasi penimbang (diberi nilai kecil secara acak)

  - **Langkah 1**

    Ulangi langkah 2 hingga 9 sampai kondisi akhir iterasi dipenuhi

  - **Langkah 2**

    Untuk masing-masing pasangan data pelatihan (*training data*) lakukan langkah 3 hingga 8

- ***Umpan maju (FeedForward)***

  - **Langkah 3**

    Masing-masing unit masukan (X_i i=1….n) menerima sinyal masukkan X_i dan sinyal tersebut disebarkan ke unit-unit bagian berikutnya (unit-unit lapis tersembunyi)

  - **Langkah 4**

    Masing-masing unit di lapis tersembunyi dikalikan dengan penimbang dan dijumlahkan serta ditambah dengan *bias*nya:

$$Z\_in_j = V_{j0} + \sum_{i=1}^{n} X_i V_{ji}$$

    Kemudian dihitung sesuai dengan fungsi pengaktif yang digunakan:

$$Z_j = f\,(\,Z\_in_j\,)$$

    Bila yang digunakan adalah fungsi sigmoid maka bentuk fungsi tersebut adalah:

$$Z_j = \frac{1}{1 + exp^{(-z\_in_j)}}$$

Sinyal keluaran dari fungsi pengaktif tersebut dikirim ke semua unit dilapis keluaran (*unit keluaran*)

- **Langkah 5**

  Masing-masing unit keluaran (Y_k , k=1,2,3….m) dikalikan dengan penimbang dan dijumlahkan serta ditambah dengan biasnya:

$$Y\_in_k = W_{k0} + \sum_{j=1}^{p} Z_j W_{kj}$$

Kemudian dihitung kembali sesuai dengan fungsi pengaktif.

$$y_k = f(y\_in_k)$$

➢ ***Backpropagasi dan Galatnya***

- **Langkah 6**

  Masing-masing unit keluaran (Y_k, k=1,…m) menerima pola target sesuai dengan pola masukan saat pelatihan/training data dihitung galatnya:

$$\delta_k = (t_k - y_k) f'(y\_in_k)$$

Karena f'(y_in_k) = y_k menggunakan **fungsi sigmoid** , maka:

$$f'(y\_in_k) = f(y\_in_k)\,(1 - f'(y\_in_k))$$

$$= y_k\,(1 - y_k)$$

Menghitung perbaikan penimbang (kemudian untuk memperbaiki w_{jk}).

$$\Delta W_{kj} = \alpha \, . \, \partial_{k..} Z_j$$

Menghitung perbaikan koreksi:

$$\Delta W_{0j} = \alpha \, . \, \partial_{k.}$$

Dan menggunakan nilai delta (d_k) pada semua unit lapis sebelumnya.

- **Langkah 7**

  Masing-masing penimbang yang menghubungkan unit-unit lapis keluaran dengan unit-unit pada lapis tersembunyi (Z_j , j=1,….p) dikalikan delta () dab dijumlahkan sebagai masukan ke unit-unit lapis berikutnya.

$$\delta\_in_j = \sum_{k=1}^{m} \delta_i W_{ij}$$

Selanjutnya dikalikan dengan turunan dari fungsi pengaktifnya untuk mengitung galat.

$$\delta_j = \delta\_in_j \, f'(y\_in_j)$$

Langkah berikutnya mengitung perbaikan penimbang (digunakan untuk memperbaiki V_{ij}).

$$\Delta V_{ij} = \alpha \delta_j \, . \, X_i$$

Kemudian menghitung perbaikan bias (untuk memperbaiki V_{oj})

$$\Delta V_{0j} = \alpha \delta_j$$

➢ ***Memperbaiki penimbang dan bias***

- **Langkah 8**

  Masing-masing keluaran unit (Y_k, k=1,….,m) diperbaiki bias dan penimbangnya (j=0,….,p),

  $$W_{kj} \text{ (baru)} = W_{kj} \text{ (lama)} + \Delta W_{kj}$$

  Masing-masing unit tersembunyi (Z_j, j=1,…,p) diperbaiki bias dan penimbangnya (j=0,…..,n).

  $$V_{ji} \text{ (baru)} = V_{ji} \text{ (lama)} + \Delta V_{ji}$$

- **Langkah 9**

  Uji kondisi pemberhentian dengan menggunakan dua cara yaitu membatasi iterasi dan menentukan besar Mean Square Error antara output yang dikehendaki dan output yang dihasilkan.

  $MSE=0,5 \times \{(t_{k1}-y_{k1})^2+ (t_{k2}-y_{k2})^2+\ldots+ (t_{km}-y_{km})^2\}$

  Daftar Notasi :

  X^p = Pola masukan pelatihan ke-p, p=1,2,……, p <= 1

  X^p = $(X_1, X_2, X_3,\ldots,X_n)$

t^p = pola target keluaran dari pelatihan

t^p = $(t_1,t_2,t_3,\ldots,t_n)$

xi = Unit ke-i pada lapisan masukan

Xi = nilai pengaktif dari unit Xi

z_j = Unit ke-j pada lapisan tersembunyi

Z_in_j = keluaran untuk unit Z_j

Z_j = nilai pengaktif dari unit Zi

Y_k = Unit ke-j pada lapisan keluaran

Y_in_k = keluaran untuk unit Y_k

y_k = nilai pengaktif dari unit Y_k

W_{k0} = nilai penimbang pada bias untuk unit Y_k

W_{kj} = nilai penimbang dari Z_{ij} ke unit Y_k

ΔW_{kj} = selisih antara W_{kj} (t) dengan W_{kj} (t+1)

V_{j0} = nilai penimbang pada bias untuk unit Z_j

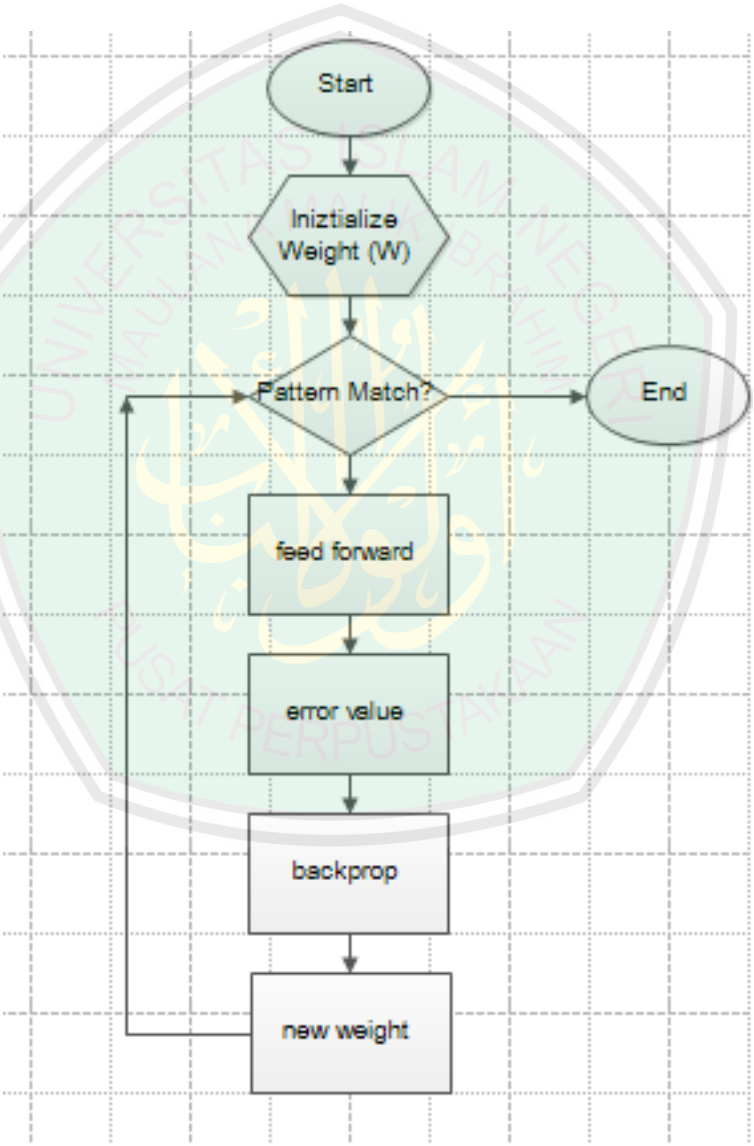
V_{ij} = nilai penimbang dari unit X_i ke unit Z_j

ΔV_{ij} = selisih antara V_{ij} (t) dengan V_{ij} (t+1)

δ_k = faktor pengendalian nilai penimbang pada lapis keluaran

δ_j = faktor pengendalian nilai penimbang pada lapis tersembunyi

α = konstanta laju pelatihan (learning rate) 0<a<1



Gambar 2. 6 Flowchart Neural Network Backpropagation

Alur kerja *Backpropagation* yang lebih sederhana dijelaskan dalam situs *www.nnwj.de* oleh Jochen Fröhlich. Ada 4 tahap yang diringkas dari langkah-langkah diatas (Fröhlich, 2015), yakni:

1. Lakukan fase *forwardpropagation* pada sebuah pola inputan dan hitung *output error*-nya
2. Ganti semua nilai bobot pada setiap matriks bobot menggunakan formula berikut:

   Formula = weight (old) + learning rate * output error * output (neurons i) * output (neurons i+1) * (1 - output(neurons i+1))

3. Kembali ke step 1
4. Algoritma berakhir, jika semua pola keluaran sesuai dengan pola target

Jika semua nilai sebuah pola inputan adalah nol, maka bobot-bobot yang ada di matriks bobot awal tidak akan berganti untuk pola ini dan proses *learn/train* pada jaringan tidak akan berhasil. Maka dibuatlah sebuah input bias yang mempunyai nilai konstanta 1 seperti yang ada pada Gambar 5.

1.5 Inisalisasi Bobot Nguyen-Widrow

Masalah yang biasanya timbul ketika proses *training* lamanya iterasi yang dilakukan. Sebenarnya dalam proses *training* sudah ditentukan *stop condition* dan iterasi maksimal yang mungkin dilakukan dalam proses tersebut.

Bobot awal dari masing-masing inputan akan mempengaruhi jaringan mencapai titik minimum local atau global, atau seberapa cepat konvergensinya. Bobot yang menghasilkan nilai turunan aktivasi yang kecil sedapat mungkin

dihindari karena akan menyebabkan perubahan bobotnya menjadi sangat kecil. Demikian pula nilai bobot awal tidak boleh terlalu besar karena nilai turunan fungsi aktivasinya menjadi sangat kecil juga. Oleh karena itu dalam standar Backpropagation, bobot dan bias diisi dengan bilangan acak kecil. (Siang, 2005)

Nguyen dan Widrow (1990) mengusulkan cara membuat inisialisasi bobot dan bias ke unit tersembunyi sehingga menghasilkan iterasi lebih cepat

Misal,

n = jumlah unit masukkan

p = jumlah unit tersembunyi

β = factor skala = $0.7\sqrt[n]{p}$

Algoritma inisialisasi Nguyen Widrow adalah sebagai berikut :

a. Inisialisasi semua bobot (v_{ji} (lama)) dengan bilangan acak dalam interval [-0.5, 0.5]
b. Hitung $\|v_j\| = \sqrt{Vj1^2 + Vj2^2 + \cdots + Vjn^2}$
c. Bobot yang dipakai sebagai inisialisasi = $V_{ji} = \frac{\beta\ vji(lama)}{||vj||}$
d. Bias yang dipakai sebagai inisialisasi = v_{j0} = bilangan acak antara $-\beta$ *dan* β

1.6 Momentum

Dewasa ini sudah berkembang beberapa variasi dari algoritma Jarisan Syaraf Tiruan *Backpropagation.* Tujuannya tidak lain adalah untuk mempercepat proses pelatihan untuk memperoleh bobot yang cocok dalam kasus-kasus tertentu.

Menurut Siang (2005: 113) pada standar *Backpropagation*, perubahan bobot didasarkan atas gradient yang terjadi untuk pola yang dimasukkan saat itu. Modifikasi yang dapat dilakukan adalah melakukan perubahan bobot yang didasarkan atas arah gradient pola terakhir dan pola sebelumnya (disebut *momentum*) yang dimasukkan. Jadi di sini bukan hanya pola masukan terakhir saja yang diperhitungkan.

Penambahan momentum ini dimaksudkan untuk menghindari perubahan bobot yang mencolok akibat adanya data yang sangat berbeda dengan yang lain (*outlier*). Apabila beberapa data terakhir yang diberikan ke jaringan memiliki pola yang serupa (berarti arah gradient sudah benar), maka perubahan bobot dilakukan secara cepat. Namun apabila data terakhir yang dimasukkan memiliki pola yang berbeda dengan pola sebelumnya, maka perubahan dilakukan secara lambat.

Dengan penambahan ini, maka bobot yang baru yakni pada waktu $(t+1)$ didasarkan atas bobot pada waktu t dan $(t-1)$. Di sini harus ditambahkan 2 variabel baru yang mencatat besarnya momentum untuk 2 iterasi terakhir. Jika μ adalah konstanta $(0 \leq \mu \leq 1)$ yang menyatakan parameter momentum maka bobot baru dihitung berdasarkan persamaan berikut:

$$w_{kj}(t+1) = w_{kj}(t) + \alpha \delta z_i + \mu(w_{kj}(t) - w_{kj}(t-1))$$

dan

$$v_{kj}(t+1) = v_{kj}(t) + \alpha \delta x_i + \mu(v_{kj}(t) - v_{kj}(t-1))$$

1.7 Bahasa Pemrograman Java

Java adalah bahasa pemrograman yang *multi-platform* dan *multi-device*. Sekali menuliskan program dengan menggunakan Java, maka aplikasi dapat dijalankan hamper pada semua computer dan perangkat lain yang support dengan Java, dengan sedikit perubahan atau tanpa perubahan sama sekali dalam kodenya. Aplikasi dengan berbasis java ini dikompulasikan ke dalam *p-code* dan bisa dijalankan dengan *Java Virtual Machine.* (Vicky, 2012)

Berikut adalah Java yang berhasil dirilis ke dunia pemrograman yang dilangsir oleh oracle.com :

- Java SE 1.1
- Java SE 1.2
- Java SE 1.3
- Java SE 1. 4
- Java SE 5
- Java SE 6 (December 11, 2006)
- Java SE 7 (July 28, 2011)
- Java SE 8 (March 18, 2014)

1.8 *Platform Android*

Android adalah Sistem Operasi yang berbasis Linux untuk telepon seluler seperti telepon pintar dan komputer tablet. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli Android Inc.,

pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengambangkan android, dibentuklah Open Handset Alliance, konsorium dari 34 perusahaaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan NVIDIA (Aingindra, 2013).

1.9 Huruf Hijaiyah

Sistem ortografi bahasa Arab memakai system Abjad. Sistem Abjad yaitu system tulisan yang huruf-hurufnyaa melambangkan bunyi konsonan sedangkan bunyi vocal dilambangkan dengan harokat. Huruf hijaiyah terdiri dari 29 huruf Abjad : 26 berupa konsonan murni dan 3 berupa konsonan semi vocal yaitu "Alif", "Wau", dan "Ya". Bunyi vocal tidak dilambangkan dengan Abjad tetapi dengan harokat. Ada 3 harokat dalam bahasa Arab yakni *Fathah* yang dilambangkan dengan bunyi "a" dan di beberapa Abjad dengan bunyi "o", *Kasroh* yang dilambangkan dengan bunyi "i", dan *Dhammah,* yang dilambangkan dengan bunyi "u".

Gambar 2. 7 Huruf Ta' (salah satu huruf hijaiyah)

1.10 OpenGL ES

Android memasukan dukuan untuk grafik 2D dan 3D yakni dengan Open Graphics Library (OpenGL), khususnya OpenGL ES API. OpenGL adalah kumpulan standard API (*Application Programming Interface*) yang menghubungkan software dengan *hardware* grafis untuk menampilkan gambar 2D dan 3D. OpenGL dirancang indepen terhadap system operasi, hardware, maupun bahasa pemrograman yang digunakan. Bahkan jika GPU (*Graphical Processing Unit*) tidak tersedia, openGL dapat dijalankan di atas software yang mengemulsi hardware, tentu dengan kinerja yang lebih rendah.

OpenGL ES sendiri adalah versi OpenGL untuk embedded system dan mobile device khususnya untuk iPhone dan Android. Untuk 'merampingkan' OpenGL ES, API OpenGL yang jarang digunakan atau terlalu kompleks dibuang (Yudi , 2014).

Pada situs resmi developer android disebutkan bahwa Android mendukung beberapa versi dari API OpenGL ES:

- OpenGL ES 1.0 and 1.1 – didukung oleh Android 1.0 ke atas
- OpenGL ES 2.0 - didukung oleh Android 2.2 (API level 8) ke atas
- OpenGL ES 3.0 - didukung oleh Android 4.3 (API level 18) ke atas.
- OpenGL ES 3.1 - didukung oleh Android 5.0 (API level 21) ke atas

Dalam penelitian ini OpenGL dimanfaarkan untuk menggambar object 2D yang bisa di atur posisi dan rotasinya sesuai koordinat xyz, agar lebih menarik.

BAB III

ANALISIS DAN PERANCANGAN

1.1 Desain Penelitian

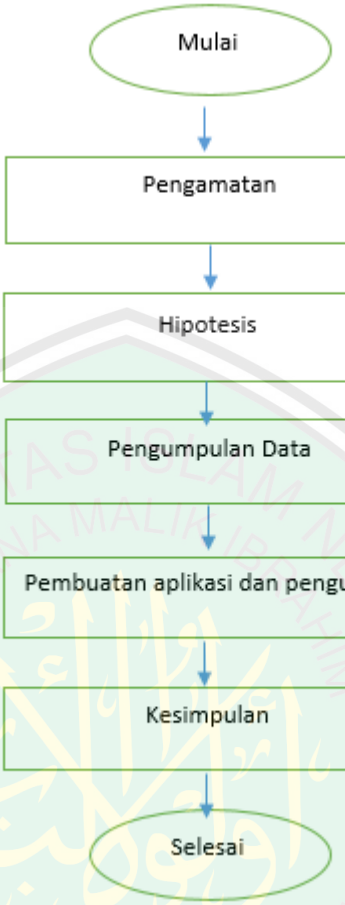
Penelitian ini menggunakan metode penelitian kuantitatif, inferensial, dan empirik. Metode analisa data kuantitif digunakan karena dalam pelaksanaannya meliputi data yang berbentuk angka atau data yang diangkakan. Metode kuantitatif menggunakan statistik sebagai alat analisa datanya (Silalahi, 2006:305). Statistic tersebut berfungsi meringkas data secara matematis Pada kedalaman analisanya, penelitian ini menggunakan metode inferensial karena bukan hanya menjelaskan, namun lebih cenderung melakukan analisis hubungan antar variabel dengan cara pengujian terhadap hipotesis yang sudah diambil. Selanjutnya dalam keobjektivitasan analisanya penelitian ini menggunakan metode empiric yaitu bagaimana suatu metode atau algoritma diukur kelayakannya *(visibility)* dalam menyelesaikan suatu masalah.

1.2 Sumber Data

Pada penelitian ini digunakan sumber data primer, yakni mengambil data dengan cara langsung atau bisa dikatakan data yang diperoleh dari tangan pertama. Data ini bisa diambil dari tulisan tangan 7 orang yang menuliskan 10 huruf hijaiyah sebagai data training. Maka data yang terkumpul adalah sejumlah 70 huruf.

1.3 Prosedur Penelitian

Flowchart alur penelitian yang digunakan sebagai acuan dalam melakukan peneleitian ini digambarkan pada gambar 3.1.



Gambar 3. 1 Flowchart prosedur penelitian

Tahap-tahap diatas diringkas dalam tiga tahap, yakni:

1. Tahap Persiapan

Dalam tahap persiapan ini dimulai dengan pengamatan, yakni melihat fenomena yang terjadi dengan suatu aplikasi yang sudah jadi. Selanjutnya membuat hipotesis tentang ide yang bisa dibuat ketika menggunakan teknologi tersebut (dalam hal ini teknologi *Augmented Reality*). Kemudian dilakukan proses pengumpulan data.

2. Tahap Penerapan

Pada tahap penerapan disimbolkan di dalam flowchart dengan pembuatan aplikasi dan pengujian hipotesis. Pada pembuatan aplikasi ini digunakan sebuah teknologi yang bernama *Augmented Reality* yang sudah dijelaskan di awal. Dan pada pengujian hipotesis ini dilakukan dengan menguji metode yang dipakai dalam aplikasi.

3. Tahap Evaluasi

Tahap ini merupakan tahap dimana aplikasi diuji coba kepada target penelitian dan diambil beberapa variabel yang memuat keberhasilan dan ketidakberhasilan penelitian tersebut. Target penelitian yang dipakai adalah versi Android yang berbeda.

1.4 Instrumen Penelitian

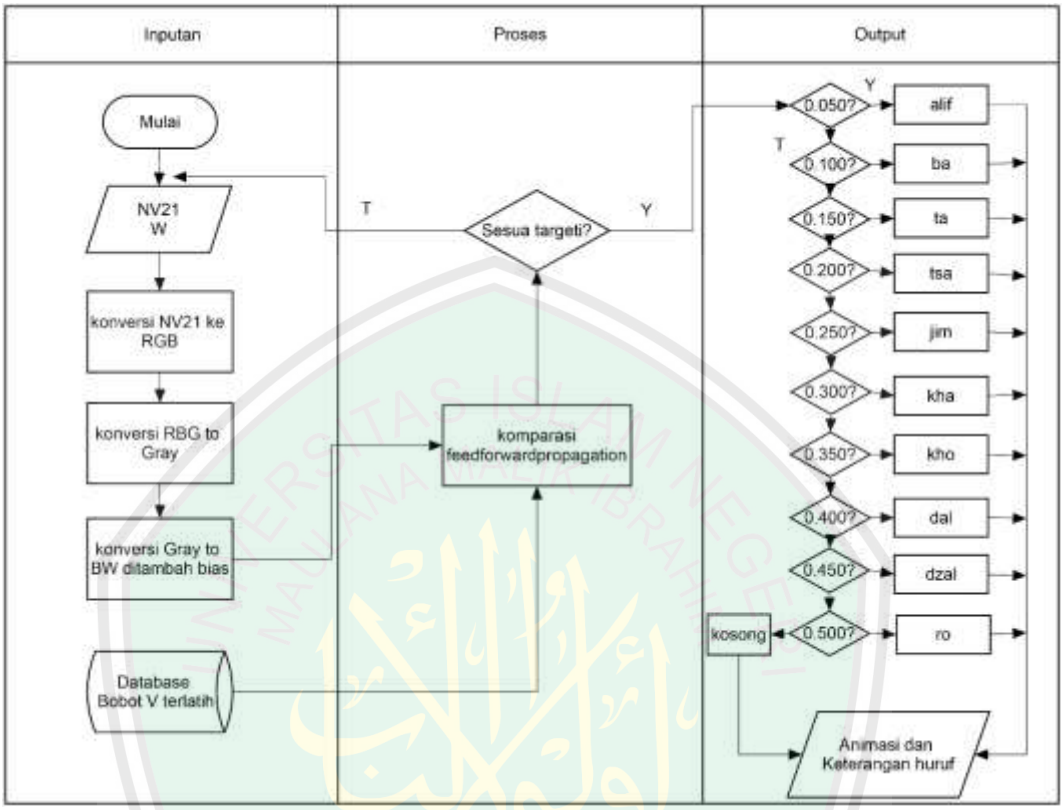
Di dalam penelitian kuantitatif, penentuan variabel adalah tahap yang penting. Variabel-variabel ini merupakan satu set parameter untuk mengukur apakah penelitian ini terdapat suatu kemanfaatan atau tidak, dan berhasil atau tidak. Di dalamnya ada tiga variabel yang digunakan dalam penelitian ini yakni variabel bebas, penghubung, dan terikat. Variabel bebas yaitu varibel yang mempengaruhi nilai dari variabel terikat. Dalam penelitian ini variabel bebas yang digunakan adalah huruf hijaiyah. Selanjutnya variabel penghubung, yakni variabel yang menghubungkan antara variabel bebas dan variabel terikat. Dalam penelitian ini, variabel penghubung yang digunakan adalah lama proses. Yang terakhir variabel terikat, yakni variabel yang muncul akibat adanya variabel bebas. Dalam penelitian ini, variabel terikat yang digunakan merupakan akurasi.

1.5 Metode Analisa Data

1.5.1 Perancangan Sistem

Aplikasi yang dibangun adalah aplikasi *Augmented Reality* untuk mengenali tulisan tangan huruf hijaiyah. Huruf hijaiyah di sini berfungsi sebagai *marker*. Dalam aplikasi ini terdapat sebuah kotak persegi berukuran 40x40 *pixel* di atas kamera perangkat android yang digunakan sebagai area untuk pembacaan gambar tulisan tangan. Objek dari aplikasi ini adalah data tulisan tangan yang diambil dari tulisan tangan 7 orang. Proses yang dilakukan dalam system pembuatannya sesuai dengan alur algoritma *Backpropagation*, yakni melalui 2 proses yakni *training* dan *testing*. *Training* dilakukan diluar aplikasi, yaitu dengan mengambil beberapa sampel data dan diproses sehingga nanti ditemukan bobot yang sesuai. *Testing* dilakukan dengan melakukan penanaman tahap *Feedforward* dari JST *Backpropagation* pada aplikasi.

Desain sistem dari proses yang ada di dalam aplikasi digambarkan pada gambar 3.2.



Gambar 3. 2 Desain Sistem Aplikasi

Bobot yang terlatih ada 2 yaitu bobot antara layer *input* dan layer *hidden* (V) dan bobot antara layer *hidden* dan layer *output* (W). Bobot W disimpan dalam database yang berupa *file* .txt karena jumlah yang sangat banyak dan tidak dimungkinkan untuk diinisialisasi di dalam pengkodean dan dipanggil ketika proses *testing* pada *camera view* dimulai. Berbeda dengan bobot W yang jumlahnya mengikuti jumlah layer *hidden* ditambah bobot bias, maka bisa diinisialisasi di dalam pengkodean. Pada gambar di atas terlihat bahwa yang menjadi data *input* untuk masuk ke proses perhitungan JST adalah piksel *Black and White* ditambah bias dan bobot V dan W. Ketika ditemukan target ataupun tidak ditemukan, proses

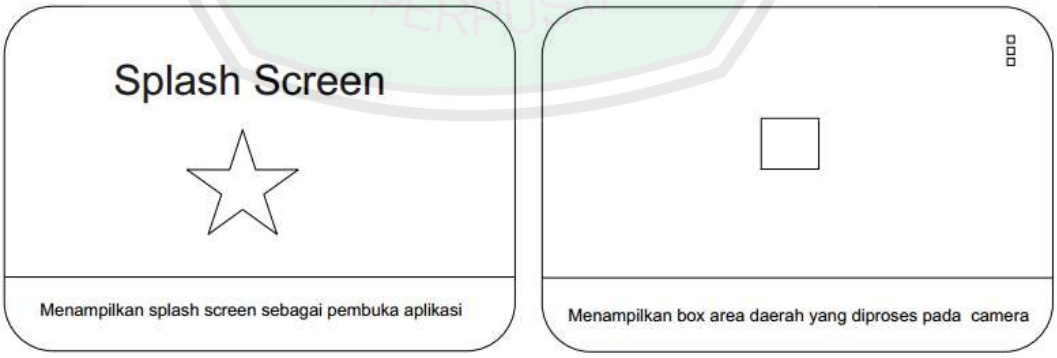
perhitungan terus berlangsung. Proses dari perolehan bobot terlatih V dan W akan dibahas pada proses *training* pada sub bab selanjutnya.

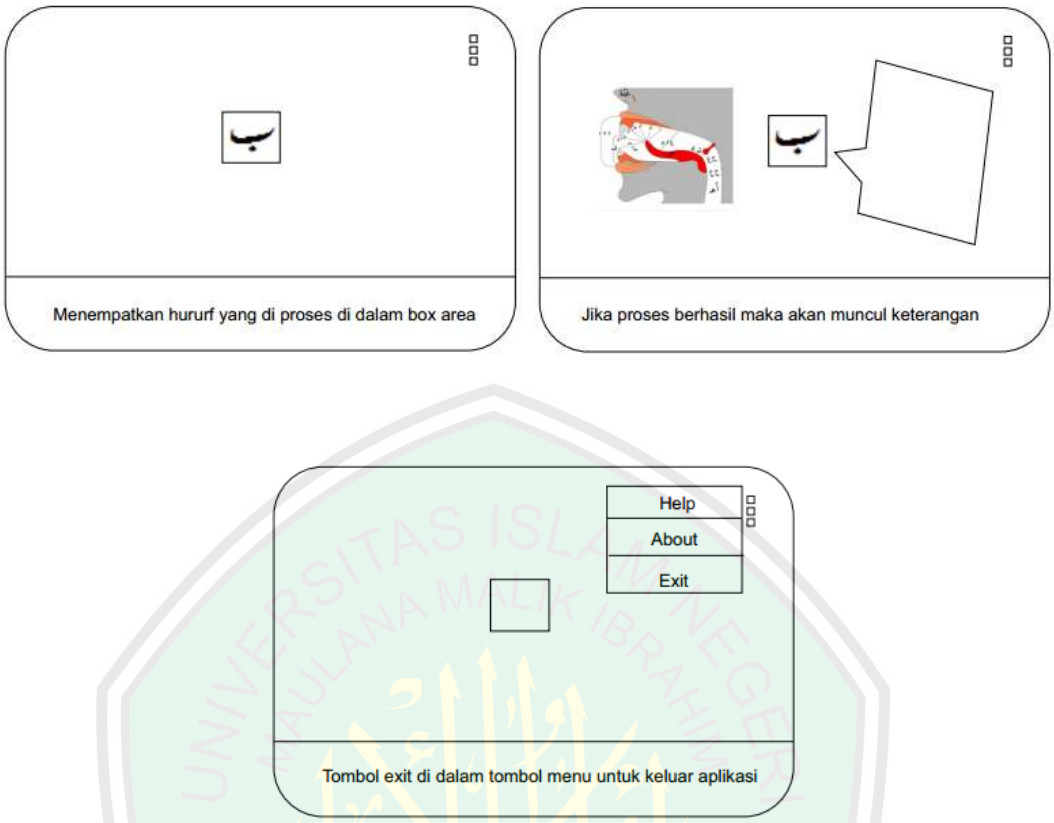
1.5.1.1 Keterangan Umum Aplikasi

Aplikasi ini berbentuk program *Augmented Reality* yang ditujukan untuk pengenalan tulisan tangan huruf hijaiyah yang diharapkan ketika pengenalan seseorang bisa langsung menulis sendiri huruf hijaiyah yang hendak dibaca. Tentunya dalam proses *testing* nanti tidak semua tulisan tangan bisa dikenali. Hal ini mungkin terjadi karena data sampel tulisan tangan yang di *training* hanya pada beberapa orang. Selanjutnya, jika tulisan tangan dikenali oleh sistem, maka aplikasi akan memunculkan keterangan dari huruf tersebut dan juga animasi bagaimana cara mengucapkan huruf atau biasa disebut dengan *makhorijul huruf* sebagai unsur edukasinya.

1.5.1.2 Storyboard Aplikasi

Beberapa perancangan *storyboard* dari aplikasi *Augmented Realiy* ini digambarkan pada gambar 3.3.





Gambar 3. 3 *Storyboard* Aplikasi

1.5.1.3Penampilan Umum Aplikasi

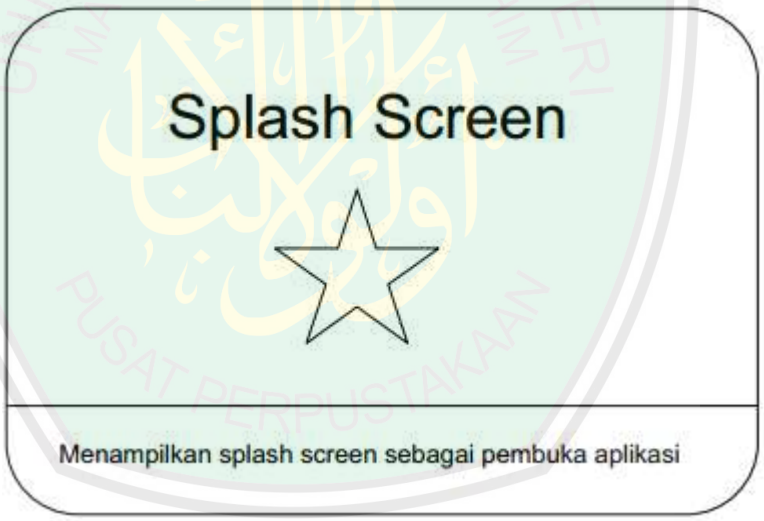
Aplikasi yang dibangun secara umum adalah aplikasi yang menampilkan animasi gif dan keterangan dari huruf hijaiyah jika system mengenali gambar. Aplikasi gif disini merupakan animasi yang menirukan cara pengucapan atau pelafalan dari huruf hijaiyah tersebut. Keterangan yang ditampilkan adalah tampilan dari *OpenGL* yang dilapisi dengan texture berupa keterangan dari huruf. *OpenGL* digunakan agar bisa mengatur tata letak dan posisi keterangan dari huruf sesuai dengan koordinat *xyz*. Secara umum tampilan yang terlihat dalam aplikasi ini adalah 2Dimensi meskipun menggunakan *OpenGL* 3Dimensi untuk pembuatan keterangan huruf.

1.5.2 Perancangan Aplikasi

Berikut ini penjelasan tentang perancangan aplikasi *Augmented Reality* berupa diagram blok dan keterangannya.

1.5.2.1 Perancangan Antarmuka Intro

Pada saat pertama kali aplikasi dijalankan akan muncul *splash screen* yang memperlihatkan logo dari aplikasi. *Splashscreen* ini akan muncul kira-kira 5 detik. Setelah itu aplikasi akan menuju antarmuka yang memperlihatkan sebuah *box area* yang digunakan sebagai daerah yang diproses dalam aplikasi.

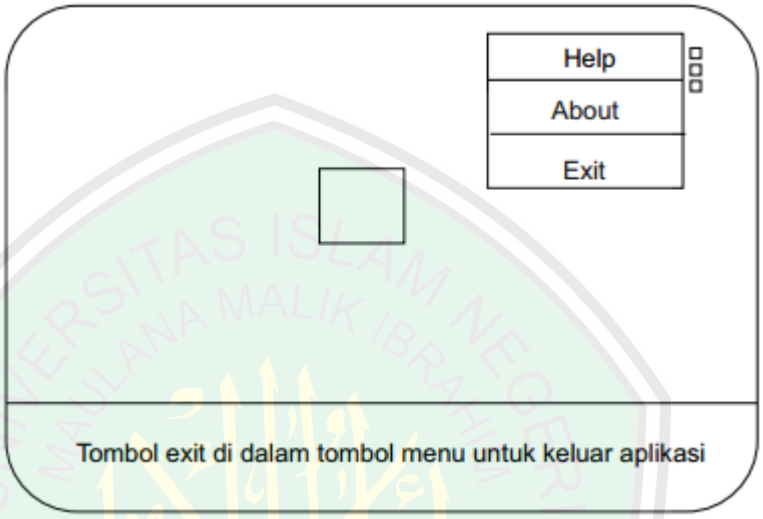


Gambar 3. 4 Antarmuka Intro aplikasi

1.5.2.2Perancangan Antarmuka Aplikasi

Setelah melewati *splash screen* aplikasi akan masuk pada antarmuka awal dengan box area berada di tengah-tengah tampilan kamera. Di sebelah kanan atas juga terdapat menu yang di dalamnya terdapat tombol-tombol:

1. Tombol Help, berfungsi untuk menunjukkan ke *user* bagaimana cara menggunakan aplikasi secara benar.
2. Tombol About, memperlihatkan identitas personal yang membuat aplikasi.
3. Tombol Exit, berfungsi untuk keluar dari aplikasi.

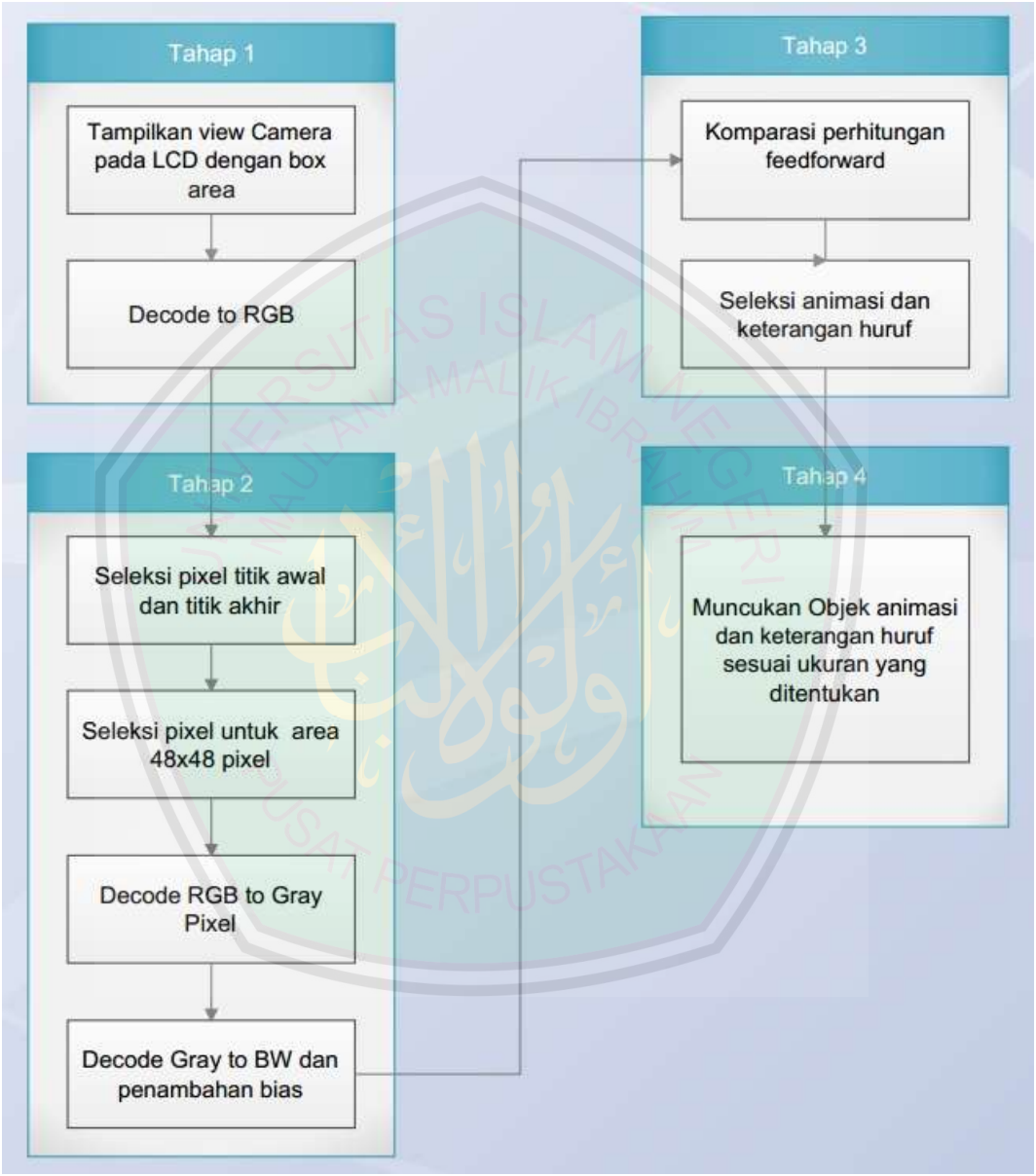


Gambar 3. 5 Antarmuka aplikasi dan menu

1.5.2.3 Perancangan Alur Aplikasi

Pada penelitian ini, *augmented reality* dikembangakn pada perangkat *mobile* android. Input yang digunakan merupakan *byte* data yang didapatkan dari kamera android dan dikonversi ke bentuk RGB array 1 dimensi menggunakan *decodeYUV420SP* dari ketai library *project* (ketai.prg). Ketai library merupakan merupakan open source library yang dikembangkan oleh Ketai (http://ket.ai/) dan Daniel Sauter (http://danielsauter.com/). Hasil dari konversi adalah RGB yang ada dalam *array* 1 dimensi dengan panjang pixel sesuai resolusi dari LCD perangkat *mobile*. Setelah itu, pixel diseleksi agar yang di proses nantinya pixel yang berada dalam *box area* dengan ukuran 40x40 pixel. Kemudian dilakukan *preprocessing*

pada pixel yang di dapatkan sehingga menghasilkan inputan binner. Gambar 3.6 menunjukkan diagram blok aplikasi *augmented realiy* untuk pengenalan tulisan tangan huruf hijaiyah.

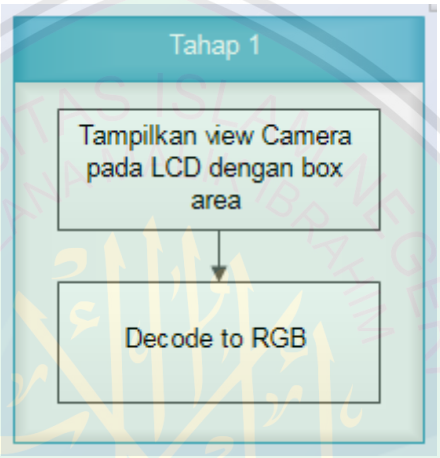


Gambar 3. 6 Diagram Blok Sistem

Berikut ini merupakan penjelasan proses untuk masing-masing tahap dalam aplikasi:

- **TAHAP 1**

Pada tahap awal ini ada 2 langkah yang dilakuakan dalam sistem, yakni menampilkan *camera view* dengan *box area* di atasnya sebagai batasan untuk memproses gambar, dan mengkonversikan data yang masuk dari *camera* berupa data *NV21* menjadi data RGB satu *array*. Diagram blok untuk Tahap pertama digambarkan pada gambar 3.7.

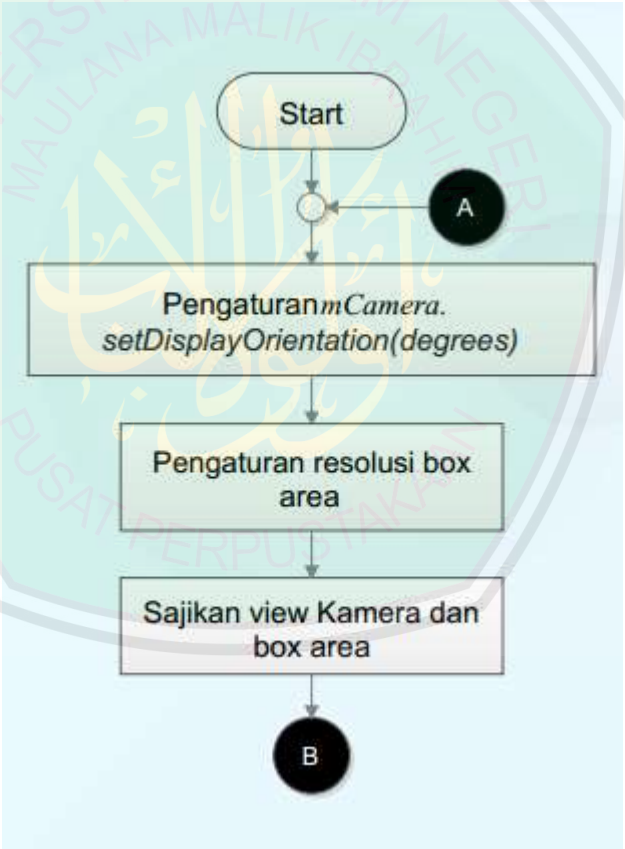


Gambar 3. 7 Diagram Blok Tahap 1

a. Tampilkan *view* kamera pada LCD dengan *box area*

Pada aplikasi ini tahap pertama setelah tampilan *splash screen* menghilang adalah penampakan *view* kamera. *View* kamera ditampilkan secara *landscape* agar proporsional dengan objek yang akan dimunculkan. Proses untuk menampilkan *view* kamera biasanya bermasalah dengan gambar yang tampil pada kamera. Masalah ini muncul karena perbandingan resolusi kamera dan resolusi LCD kebanyakan tidak sama, sehingga gambar yang ditampilkan dalam *view* kamera perangkat biasanya akan melebar atau memanjang. Dalam *Stackoverflow* (stackoverflow.com) bahwa masalah ini pernah menjadi *bug issue*. Namun masalah

tersebut sebenarnya bisa diselesaikan dengan menggunakan *mCamera. setDisplayOrientation(degrees)* yang tersedia di API 8. *DroidAR Framework* (bitstars.com) juga menggunakan cara ini dalam pembuatan *AR framework project*-nya. Namun menurut pembuatanya cara tersebut tidak berjalan pada *older device* atau perangkat yang lama. Setelah kamera terpasang, maka selanjutnya adalah *view* untuk *box area*-nya. *Box area* dibuat dengan menggunakan *Canvas*, dengan ukuran 40x40 pixel sesuai dengan pixel image yang akan diproses. *Flowchart* untuk *view* kamera pada LCD perangkat digambarkan pada gambar 3.8.



Gambar 3. 8 Flowchart untuk menampilkan view kamera dan box area

Kode sumber untuk menampilkan view kamera dan box area pada LCD perangka digambarkan pada gambar 3.9.

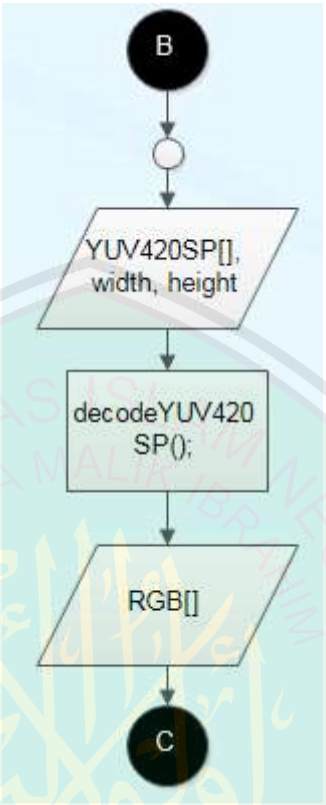
```
CameraView cameraView = newCameraView(this);
BoxArea boxArea = new BoxArea(this);
setContentView(cameraView);
addContentView(glSurfaceView,new
WindowManager.LayoutParams(WindowManager.LayoutParams.WRAP_CON
TENT,WindowManager.LayoutParams.WRAP_CONTENT));
addContentView(boxArea,new
WindowManager.LayoutParams(WindowManager.LayoutParams.WRAP_CON
TENT,WindowManager.LayoutParams.WRAP_CONTENT));
```

Gambar 3. 9 Kode sumber untuk menampilkan view kamera dan box area

b. *Decode pixel to RGB*

Input yang digunakan dalam proses komparasi berupa bilangan 0 dan 1 yang mewakili hitam dan putih. Namun sebelumnya, ada beberapa tahap yang harus dilalui, salah satunya adalah mengkonversi data inputan dari *camera device* ke bentuk RGB. Inputan dari *camera device* merupakan *byte data* dalam *array* 1 dimesi berupa format *NV21* yang kemudian dikonversi dengan *method* yang dipakai dalam *ketai project (*ketai.prg*)* ke bentuk data *Integer* RGB. Di dalam proses pembuatan dan percobaan, digunakan perangkat *mobile* Asus Zenfone 4 yang mempunyai kamera dengan besar 5 Megapixel dengan resolusi LCD 480x800 *pixels*. Panjang atau *length* dari inputan pertama yaitu 576000. Setelah dikonversi ke bentuk RGB, maka *length*-nya menjadi 384000.

Flowchart untuk proses konversi dari *NV21* ke bentuk RGB dengan array 1 dimensi digambarkan pada gambar 3.10.

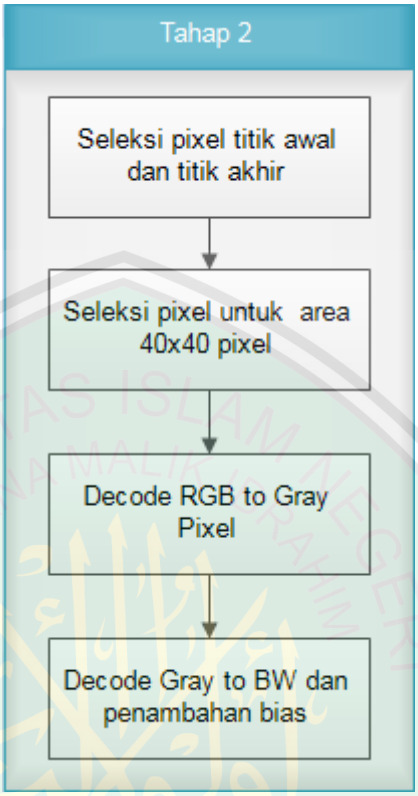


Gambar 3. 10 Flowchart proses konversi ke bentuk RGB 1 dimensi

- **TAHAP 2**

Pada tahap yang kedua ini proses yang dilakukan terbagi menjadi 4 langkah yakni menyeleksi titik awal dan titik akhir dari titik *pixel* yang diambil, menyeleksi *pixel* yang berada di daerah 40x40 *pixels* dari seleksi awal, mengkonversikan data RGB yang sudah dipotong dalam bentuk *array Grayscale*, dan yang terakhir adalah mengkonversikan data *pixel GrayScale* ke dalam bentuk Black and White yang selanjutnya langung di tambah nilai

bias pada index ke-0. Diagram blok untuk Tahap yang ke-2 digambarkan pada gambar 3.11.



Gambar 3. 11 Diagram Blok Tahap 2

a. Seleksi pixel dari titik awal dan titik akhir *box area*

Setelah melalui proses konversi, pixel yang dihasilkan adalah pixel sejumlah ukuran layar LCD dari perangkat. Kebutuhan inputan system hanya pada pixel yang terletak di dalam *box area.* Maka yang dilakukan selanjutnya adalah penyeleksian titik awal dan titik akhir dari *pixel* yang ada di *box area.*

Penyeleksian titik awal dan titik akhir harus melalui perhitungan karena untuk mengantisipasi perbedaan ukuran LCD pada beberapa kamera *mobile devic*e. Namun seperti yang diketahui bahwa panjang dan lebar dari LCD *mobile device* adalah bilangan genap.

Nexus S	4.0"	480x800	hdpi
Nexus One	3.7"	480x800	hdpi
Nexus 6	5.96"	1440x2560	560dpi
Nexus 5	4.95"	1080x1920	xxhdpi
Nexus 4	4.7"	768x1280	xhdpi
Galaxy Nexus	4.65"	720x1280	xhdpi
5.4" FWVGA	5.4"	480x854	mdpi
5.1" WVGA	5.1"	480x800	mdpi
4.7" WXGA	4.7"	720x1280	xhdpi

Gambar 3. 12 Beberapa resolusi LCD pada AVD manager Android Studio

Contoh penyeleksian titik awal dan titik akhir dari *box area yang* berada di tengah-tengah layar LCD digambarkan pada gambar 3.13.

1		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2		17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
3		33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
4		49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
5		65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
6		81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96
7		97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112
8		113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Gambar 3. 13 Contoh pixel LCD dengan resolusi 8x16 pixels

Dimisalkan hasil konversi kamera android berupa *pixel* RGB dengan resolusi 8x16 *pixels* dengan *box area* seberas 4x4 *pixels.* Maka formula yang digunakan digambarkan pada gambar 3.14.

Titik awal = (W x Jarak atas) + ((W/2)- (Tengah Pixel-1))

Titik akhir = (((H - Jarak atas) x W) - ((W-pixel)/2))

Keterangan :

W = lebar

H = tinggi

Jarak atas = (H - pixel)/2, merupakan jarak dari atas ke *box area*

Pixel = 4, sesuai panjang pixel dengan catatan *box area* adalah persegi

Tengah Pixel = Pixel/2

Gambar 3. 14 Cara menghitung titik awal dan titik akhir

Perhitungan di atas tentu saja tidak bisa langsung digunakan karena index dari array dimulai dari angka 0, jadi masing-masing titik harus dikurangi dengan angka 1.

Maka, jika dilakukan perhitungan pada *device* Android dengan resolusi 480x800 dengan 480 sebagai tinggi dan 800 sebagai lebar maka perhitungan akan menjadi sebagai beriku :

Pixel = 40

Jarak atas = (480-40)/2 = 220

Tengah pixel = 40/2 = 20

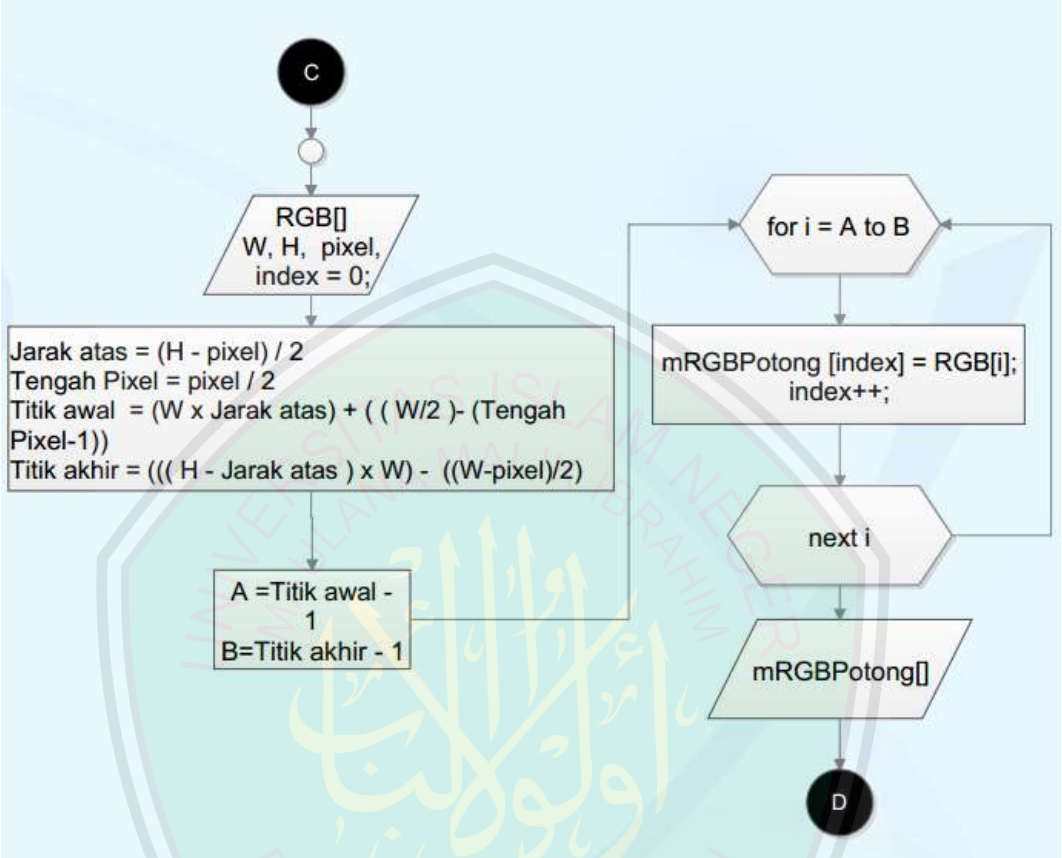
Titik awal = (800 x 220) + ((800/2)- (20-1))

Titik akhir = (((480 - 220) x 800) - ((800-40)/2))

Titik awal (index) = 176381 – 1 = 176380

Titik akhir (index) = 207620 – 1 = 207619

Flowchart dari penyeleksian titik awal dan titik akhir digambarkan pada gambar 3.15.



Gambar 3. 15 *Flowchart* penyeleksian titik awal dan titik akhir

Huruf W dan H dalam *flowchart* merupakan lebar dan tinggi dari resolusi yang ada pada *device* Android. Perulangan akan dimulai pada angka index sesuai titik awal dan berhenti pada titik akhir. Jadi nilai RGB yang diambil adalah nilai RGB pada index-index tersebut dan disimpan dalam mRGBPotong dengan index mulai dari angka 0.

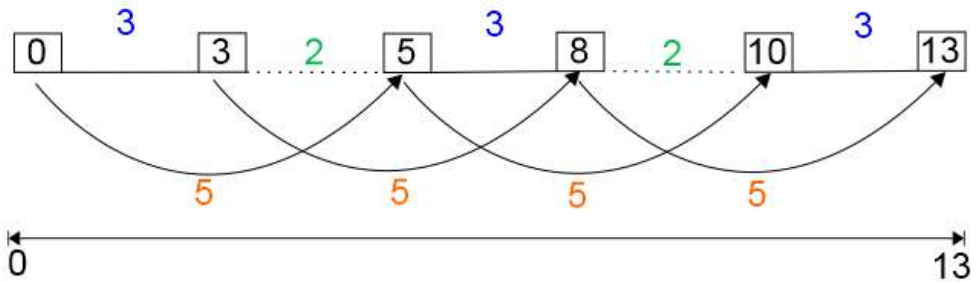
Kode sumber dari penyeleksian titik awal dan akhir digambarkan pada gambar 3.16.

```
int jarakAtas = (mImageHeight - pixel) / 2;
titikAwal = ((mImageWidth * jarakAtas) + ((mImageWidth / 2) - (tengahPix - 1))) - 1;
titikAkhir = ((mImageWidth * (mImageHeight - jarakAtas)) - ((mImageWidth - pixel) / 2)) – 1;
mRGBPotong = new int[(titikAkhir - titikAwal) + 1];
int index = 0;
for (int i = titikAwal; i <= titikAkhir; i++) {
                mRGBPotong[index] = rgb[i];
                index++;}
```

Gambar 3. 16 Kode sumber penyeleksian titik awal dan titik akhir

b. Seleksi area 40x40 *pixels*

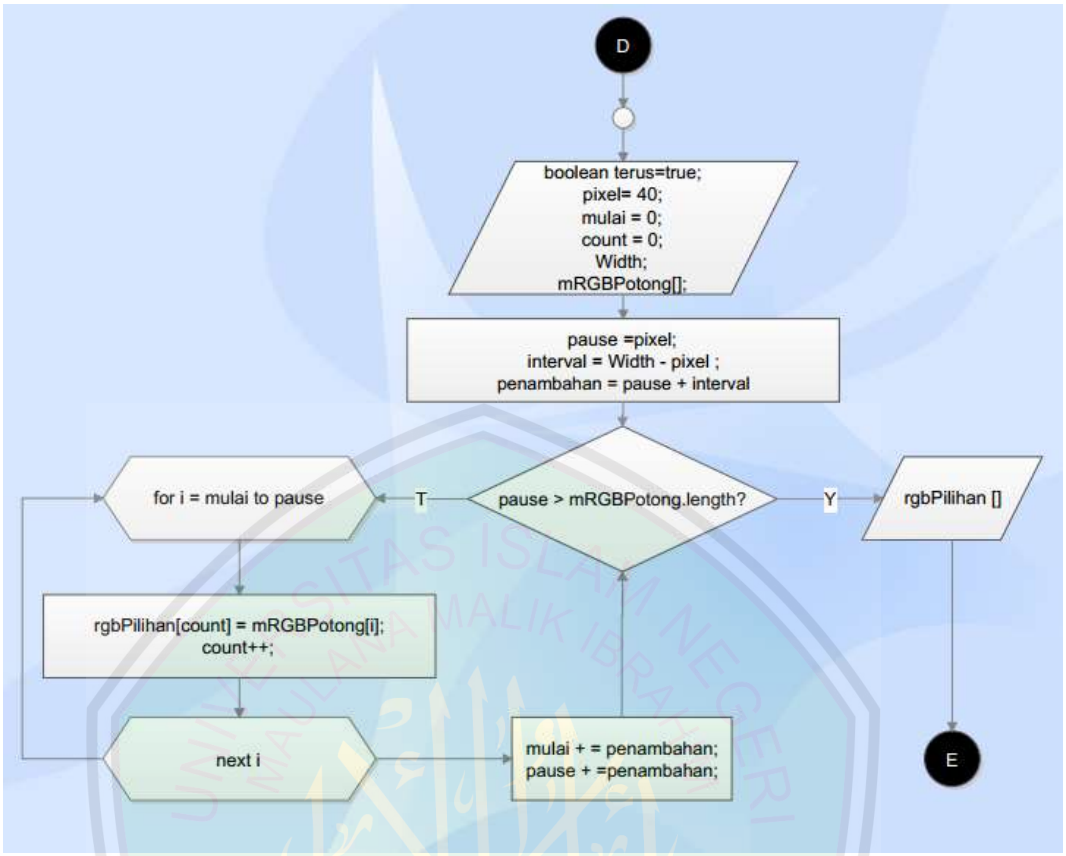
Pada tahap ini, seteleh proses sebelumnya sudah mendapatkan sejumlah pixel yang berada pada titik awal dan titik akhir dari area, maka yang dilakukan adalah seleksi untuk pixel yang berasa pada area 40x40 *pixels* saja. Penjelasan untuk penyeleksiannya digambarkan pada gambar 3.17.



Gambar 3. 17 Ilustrasi penyeleksian pixel

Area 40x40 *pixels* pada camera adalah piksel-piksel yang berada pada proses penyeleksian sebelumnya. Jika diperhatikan pada gambar 12, maka akan diketahui bahwa pixel yang dibutuhkan adalah pixel yang berada di titik awal sampai pixel ke-40 dari titik awal, kemudian pixel berikutnya akan tereliminasi sampai pada pixel yang berada pada area 40x40 *pixels*. Jika diperhatikan maka akan didapatkan pola yang sama pada gambar 15. Berdasarkan gambar 15 titik awal adalah angka 0, dan titik akhir adalah angka 13, pixel yang diambil adalah 3x3 *pixels*, angka 2 adalah jarak antara pixel ke-3 dengan pixel berikutnya yang ada di area 3x3 *pixels* (dalam contoh ini adalah angka 5). Maka perulangan dilakukan di antara angka-angka 0-3, 5-8, dan 10-13. Jadi antara batas-batas pada masing perulangan ada jarak 5 pixel.

Flowchart untuk pengambilan *pixel* pada area 40x40 *pixels* sesuai dengan ilustrasi contoh digambarkan pada gambar 3.18.



Gambar 3. 18 *Flowchart* seleksi area 40x40 pixels

Perulangan akan dilakukan pada *pixel* mulai index ke-0 sampai index ke-39 dan disimpan pada *array* rgbPilihan dengan index mulai dari 0. Selanjutnya *pixel* pada index berikutnya dilewati sesuai jumlah penambahan pada *index* mulai dan index *pause.* Setelah itu dicek apakah pemberhentian index *pixel* sudah lebih dari panjang mRGBPotong. Jika tidak maka perulangan akan diteruskan dimulai dari index *mulai* setelah ditambahkan dengan penambahan dan diakhiri dengan index *pause* dan disimpan dengan index lanjut dari index sebelumnya pada *array* rgbPilihan.

Kode sumber untuk penyeleksian area 40x40 *pixels* dari pixel yang di dapat dari titik awal dan titik akhir pada alur sebelumnya digambarkan pada gambar 3.19.

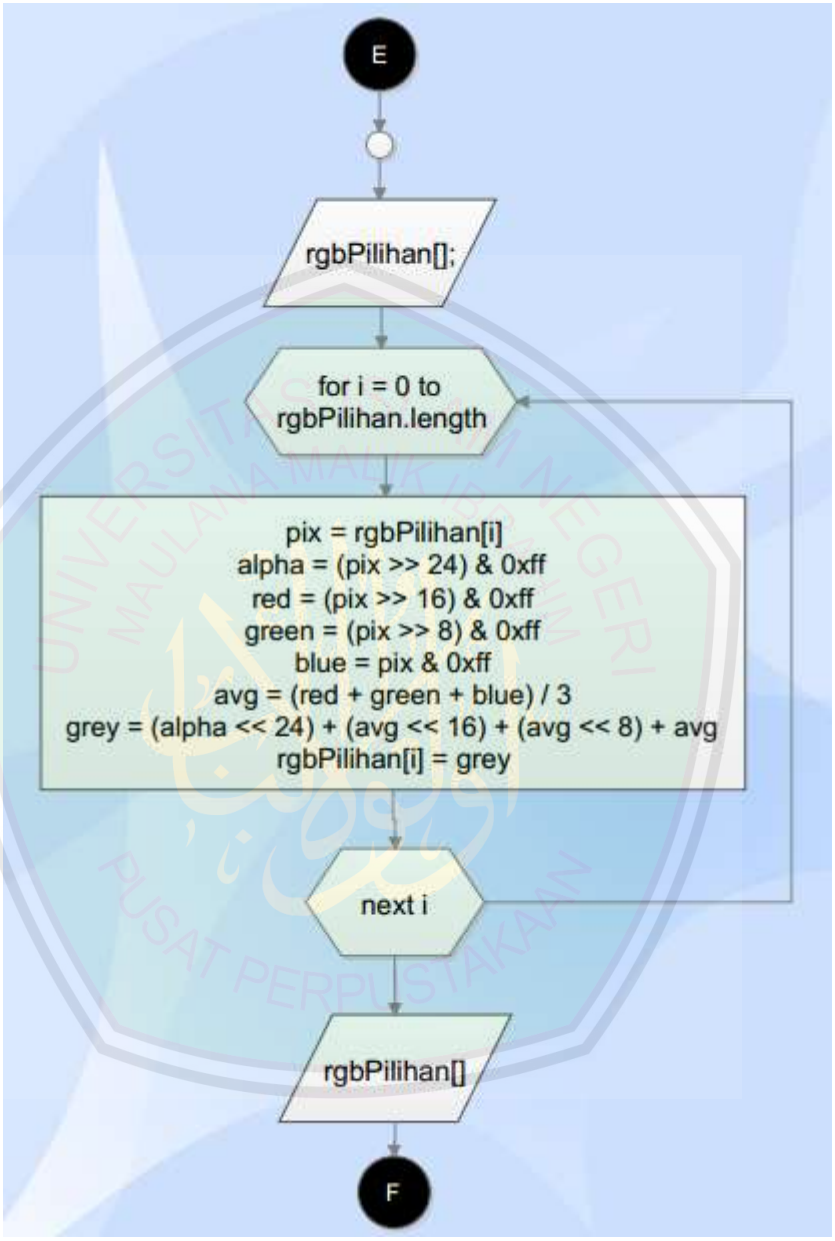
```
boolean terus = true;
        int mulai = 0;
        int pause = pixel;
        int interval = mImageWidth - pixel;
        final int penambahan = pause + interval;
        int count = 0;
        while (terus) {
           if (pause > mRGBPotong.length) {
              terus = false;
           } else {
              for (int i = mulai; i < pause; i++) {
                 rgbPilihan[count] = mRGBPotong[i];
                 count++;
              }
              mulai += penambahan;
              pause += penambahan;
           } }
```

Gambar 3. 19 Kode sumber penyeleksian area 40x40 pxiels

c. *Decode RGB* to *Gray pixels*

Tahap selanjutnya yakni mengubah piksel-piksel yang sudah terseleksi menjadi bentuk *Grayscale*. Proses ini dilakukan dengan mencari nilai rata-rata dari

pixel *RGB*. *Flowchart* untuk mengkonversi pixel-pixel RGB yang sudah kita seleksi pada area 40x40 *pixels* menjadi pixel *graysclale* digambarkan pada gambar 3.20.



Gambar 3. 20 *Flowchart* untuk mengkonversi RGB menjadi Grayscale

Kode sumber untuk mengkonversi pixel RGB menjadi piksel *grayscale* digambarkan pada gambar 3.21.

```
static public void decodeRgbToGray(int[] rgbPilihan) {
      for (int i = 0; i < rgbPilihan.length; i++) {
         int pix = rgbPilihan[i];
         int alpha = (pix >> 24) & 0xff;
         int red = (pix >> 16) & 0xff;
         int green = (pix >> 8) & 0xff;
         int blue = pix & 0xff;
         int avg = (red + green + blue) / 3;
  int grey = (alpha << 24) + (avg << 16) + (avg << 8) + avg;
         rgbPilihan[i] = grey;
      }}
//Tanda >> artinya geser
```

Gambar 3. 21 Kode sumber konversi RGB to Gray

d. *Decode Gray to Black and White* dan penambahan bias

Proses terakhir dari tahap 2 ini adalah mengkonversi pixel *Grayscale* menjadi pixel Biner (*Black and White*). Nilai *pixel* ini didapatkan dari proses *thresholding. Thresholding* adalah proses memisahkan gambar ke dalam daerah intensitasnya masing-masing sehingga bisa dibedakan antara objek dan background. Nilai *pixel Black and White* umumnya diwakili oleh 2 angka, yakni angka 1 untuk warna hitam, dan angka 0 untuk warna putih. Hanya dibutuhkan 1 *byte* untuk mewakili nilai setiap *pixel* dari gambar biner.

Cara yang digunakan untuk mengekstrak objek dari *background* adalah dengan memilih *threshold* T yang membagi node-node ini. Kemudian sembarang titik (x, y) untuk dimana f (x,y) ≥ *T* disebut *object point.* Sedangkan yang lainya disebut *background point.* Maka, gambar yang di-*thresholding* bisa didefinisikan sebagai berikut.

$$g(x,y) = \begin{cases} 1\ jika\ f(x,y) \geq T \\ 0\ jika\ f(x,y) < T \end{cases}$$

Piksel yang diberi nilai 1 berkaitan dengan objek sedangkan piksel yang diberi nilai 0 berkaitan dengan *background.*

Nilai T dapat ditentukan dengan salah satu dari 3 cara berikut (Putra, 2004).

- *Global Threshold*

  T = T{f(x,y)}, dengan T tergantung pada nilai *gray level* dari *pixel* pada posisi x,y

- *Local Threshold*

  T=T{A(x,y), f(x,y)}, dengan T tergantung pada properti *pixel* tetangga, *A(x,y)* menyatakan *pixel* tetangga.

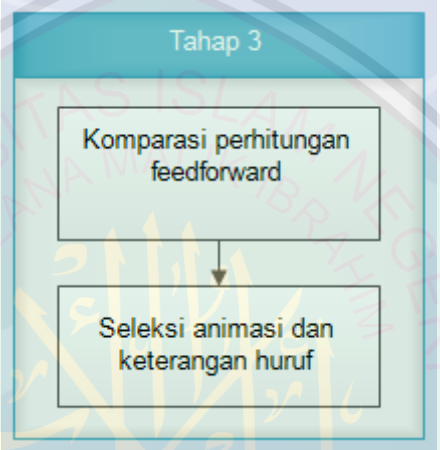
- *Dynamic Threshold*

  T=T{x,y, A(x,y), f(x,y)}, dengan T tergantung pada koordinat-kooordinat *pixel.*

Metode *Thresholding* yang digunakan dalam penlitian ini adalah metode Otsu yang termasuk dalam *Global Threshold.* Pendekatannya yang dilakukan adalah dengan menganalisis diskriminan yaitu menentukan suatu variabel yang dapat membedakan antara dua atau lebih kelompok yang muncul secara alami. Analisis diskriminan akan memaksimumkan variabel tersebut agar dapat membagi objek *foreground* dan *background.*

- **TAHAP 3**

Pada tahap yang ketiga ini proses dibagi menjadi 2 langkah yakni mengkomparasikan nilai target dari inputan *Black and White* yang dihitung dengan *feedforwardpropagatioan* dengan target yang dipakai ketika *training*, dan menyeleksi animasi dan keterangan huruf yang keluar sesuai hasil perhitungan komparasi. Diagram blok untuk tahap yang ke-3 digambarkan pada gambar 3.22.



Gambar 3. 22 Diagram Blok Tahap 3

a. Komparasi perhitungan *feedforwardpropagation*

Tahap ini merupakan tahap pengujian, dan perhitungan dilakukan di dalam perangkat android. Setelah pixel melalui beberapa proses yang disebut dengan *preprocessing*, maka jumlah pixel yang ada di dalam *array* adalah 1601, dengan penambahan 1 bias. Piksel-piksel ini selanjutnya di proses dengan *feedforwardpropagation* untuk menemukan nilai yang sesuai dengan target yang sudah ditentukan ketika proses *training.* Bobot yang digunakan dalam proses ini adalah bobot input layer dan bobot hidden layer yang sudah dilatih sebelumnya.

Kode sumber *feedforwardpropagation* digambarkan pada gambar 3.23.

```
//feedforwardpropagation
    double temp;
    double Y = 0;
    double[] Z = new double[jumlahHiddenLayer + 1];
    Z[0] = 1;
    for (int i = 0; i < jumlahHiddenLayer; i++) {
       temp = 0;
       for (int j = 0; j < mBWDataBias.length; j++) {
          temp += mBWDataBias[j] * nil[j][i];
       }
       Z[i + 1] = sigmoid(temp);
    }
    for (int j = 0; j < Z.length; j++) {
       Y += Z[j] * wHiddenLayer[j];
    }
    Y = Double.parseDouble(df.format(sigmoid(Y)));
```

Gambar 3. 23 Kode Sumber feedforwardproagation

b. Seleksi animasi dan keterangan huruf hijaiyah

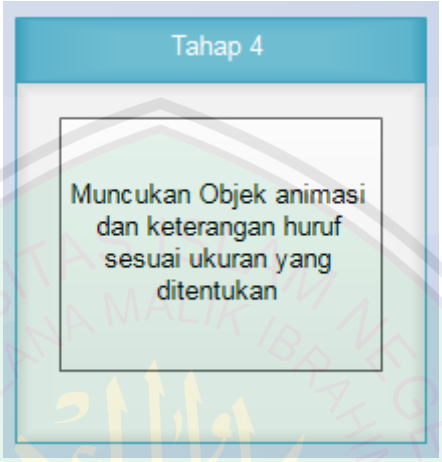
Setelah menemukan nilai yang cocok, selanjutnya yaitu proses penyeleksian animasi dan keterangan huruf hijaiyah yang sesuai dengan huruf yang di kenali. Gambar 3.24 merupakan kode sumber untuk proses penyeleksian animasi dan keterangan huruf hijaiyah.

```
if(nilTarget == 0.050 || nilTarget == 0.0501){
 is=context.getResources().openRawResource(R.drawable.alif);
      }else if(nilTarget ==0.100){
         is = context.getResources().openRawResource(R.drawable.ba);
      }else if(nilTarget ==0.150){
         is = context.getResources().openRawResource(R.drawable.ta);
      }else if(nilTarget ==0.200){
         is = context.getResources().openRawResource(R.drawable.tsa);
      }else if(nilTarget ==0.250){
         is = context.getResources().openRawResource(R.drawable.jim);
      }else if(nilTarget ==0.300){
         is = context.getResources().openRawResource(R.drawable.kha);
      }else if(nilTarget ==0.350){
         is = context.getResources().openRawResource(R.drawable.kho);
      }else if(nilTarget ==0.400){
         is = context.getResources().openRawResource(R.drawable.dal);
      }else if(nilTarget ==0.450){
         is = context.getResources().openRawResource(R.drawable.dzal);
      }else if(nilTarget ==0.500){
         is = context.getResources().openRawResource(R.drawable.ro);
      }else{ is = context.getResources().openRawResource(R.drawable.kosong);
}
```

Gambar 3. 24 Kode sumbur untuk seleksi keterangan huruf

- **TAHAP 4**

Tahap yang terakhir adalah tahap yang keempat dan mempunyai 1 langkah yakni Memunculkan objek animasi dan keterengan huruf sesuai proses seleksi pada tahap 3. Diagram blok untuk tahap yang ke-4 digambarkan pada gambar 3.25.



Gambar 3. 25 Diagram Blok Tahap 4

Proses terakhir dari system yang dibangun adalah pemunculan objek keterangan huruf dan animasi gif untuk pengucapan huruf yang dikenali. Proses ini dilakukan ketika nilai yang diharapkan ditemukan.

Kode sumber untuk pemunculan animasi dan ketengan huruf digambarkan pada gambar 3.26.

```
public void setGiv(double nilSig){
   if (gifView!=null){
    ((ViewGroup) gifView.getParent()).removeView(gifView);
    ((ViewGroup) glSurfaceView.getParent()).removeView(glSurfaceView); }
 Display display =
((WindowManager)getSystemService(Context.WINDOW_SERVICE)).getDefau
ltDisplay(); gifView = new GifView(this, nilSig);
gifView.setX((display.getWidth() / 2) - (gifView.getMovieWidth() / 2) - 140);
gifView.setY((display.getHeight() / 2) - (gifView.getMovieHeight() / 2));
glSurfaceView  = new GLSurfaceView(this);
glSurfaceView.setEGLConfigChooser(8, 8, 8, 8, 16, 0);
glSurfaceView.getHolder().setFormat(PixelFormat.RGBA_8888);
glSurfaceView.setRenderer(new GLCLearRenderer(this, nilSig));
glSurfaceView.setZOrderMediaOverlay(true);
addContentView(glSurfaceView, new
WindowManager.LayoutParams(WindowManager.LayoutParams.WRAP_CON
TENT, WindowManager.LayoutParams.WRAP_CONTENT));
addContentView(gifView, new
WindowManager.LayoutParams(WindowManager.LayoutParams.WRAP_CON
TENT, WindowManager.LayoutParams.WRAP_CONTENT));
}
```

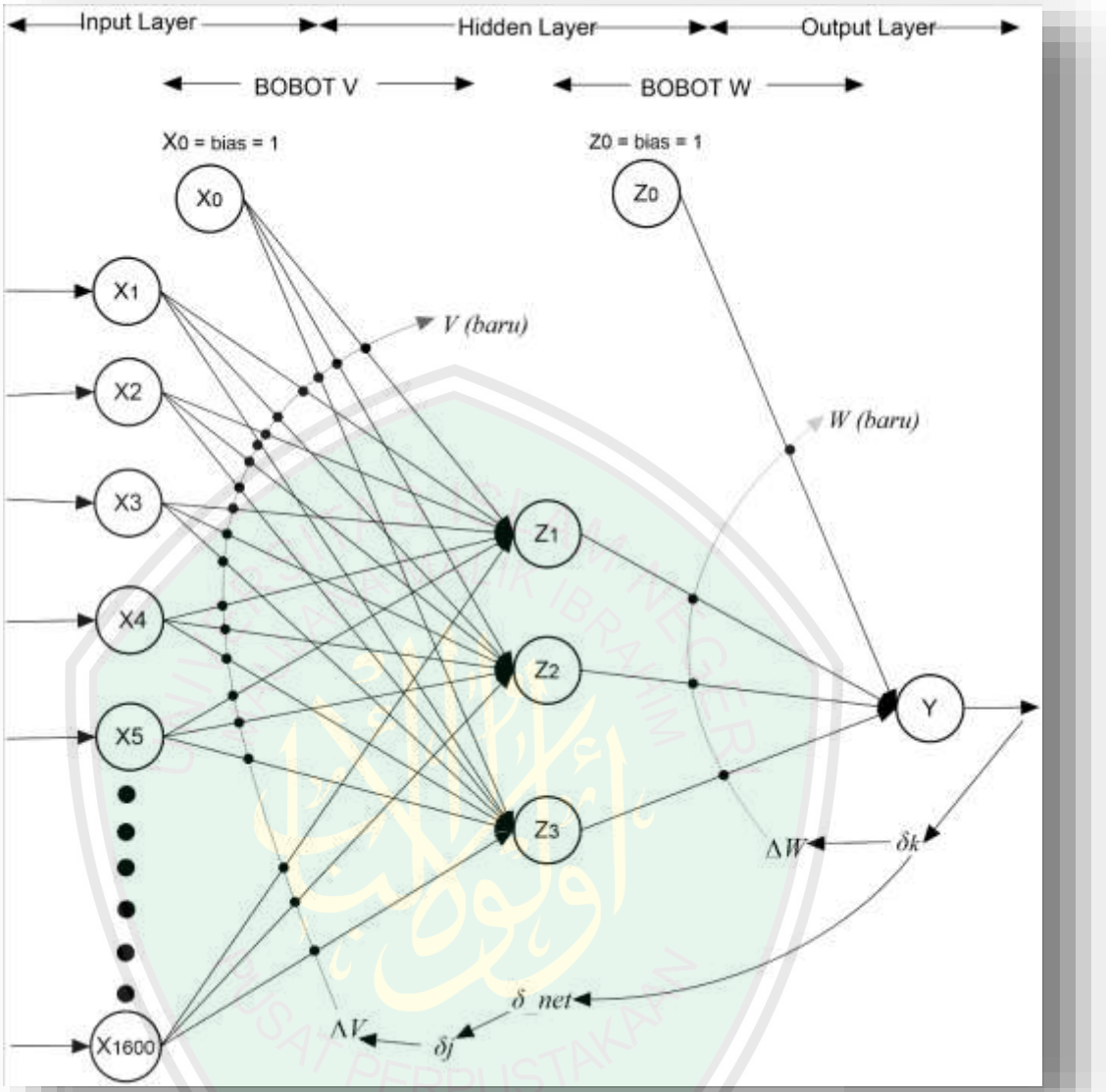
Gambar 3. 26 Kode Sumber pemunculan animasi dan keterangan huruf

1.5.3 Perancangan Algoritma

Proses perhitungan dilakukan dengan dua proses terpisah, yakni proses *training* dan proses *testing*. Proses *training* bertujuan untuk melatih bobot-bobot yang diinisialisasi secara random sesuai dengan jumlah inputan yang masuk pada *algoritma*. Sedangkan proses *testing* yakni proses untuk membuktikan bahwa bobot-bobot yang dilatih bisa membantu proses pengenalan huruf hijaiyah.

1.5.3.1 Arsitektur *Backpropagation*

Rancangan arsitektur *backpropagation* pada gambar 3. 27 dibuat berdasarkan kasus yang ada, yakni dengan inputan (X) sebanyak 1600 ditambah 1 bias, 3 buah *node* (Z) pada *hidden layer* ditambah 1 bias, dan 1 *node* pada *output layer* (Y).



Gambar 3. 27 Arsitektur *Backpropagation*

Hidden Layer

$$Z\_in_0 = 1$$

$$Z\_in_j = V_{j0} + \sum_{i=1}^{1600} X_i V_{ji}$$

$$Z_j = \frac{1}{1 + exp^{(-z\_in_j)}}$$

Output layer. Output layer hanya 1 node, maka penghitungannya menjadi:

$$Y\_in \quad = W_{10} + \sum_{j=1}^{3} Z_j W_{1j}$$

$$Y \quad = \frac{1}{1 + exp^{(-Y\_in \quad)}}$$

Fakor δ pada unit keluaran (δk). Karena jaringan hanya punya 1 keluaran maka persamaanya:

$$\delta = (t - y) y(1 - y)$$

Suku perubahan bobot (ΔW).

$$\Delta W_{kj} = \alpha \, . \, \partial_{k} \, .. Z_j$$

Penjumlahan kesalahan unit tersembunya (δ_net)

$$\delta\_net_j = \delta k \; W_{ij}$$

Fakor δ pada unit tersembunyi (δj)

$$\delta j = \delta\_net_j \; z_j \; (1 - \; z_j)$$

Suku perubahan bobot (ΔV).

$$\Delta V_{ji} = \alpha \, . \, \partial_{j} \, .. X_i$$

Perubahan bobot Wbaru dan Vbaru

$W_{kj}\ (baru) = W_{kj}\ (lama) + \Delta W_{kj}$

$V_{ji}\ (baru) = V_{ji}\ (lama) + \Delta V_{ji}$

Tanpa Momentum

$W_{kj}\ (baru) = W_{kj}\ (t) + \Delta W_{kj} + \eta(W_{kj}\ (t) - W_{kj}\ (t-1))$

$V_{ji}\ (baru) = V_{ji}\ (t) + \Delta V_{ji} + \eta(V_{ji}\ (t) - V_{ji}\ (t-1))$

Dengan Momentum

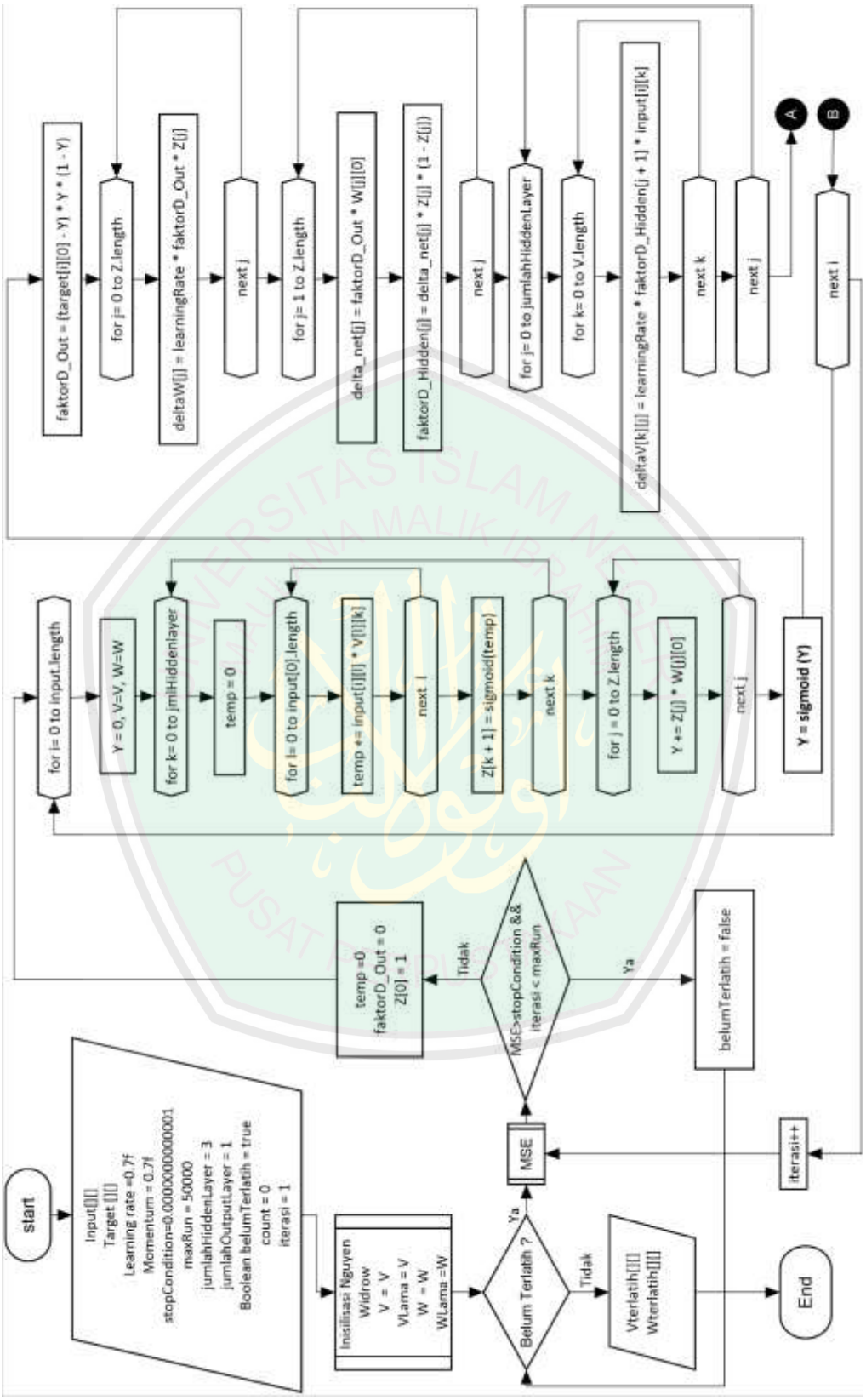
Pemberhentian *epoch* dengan MSE

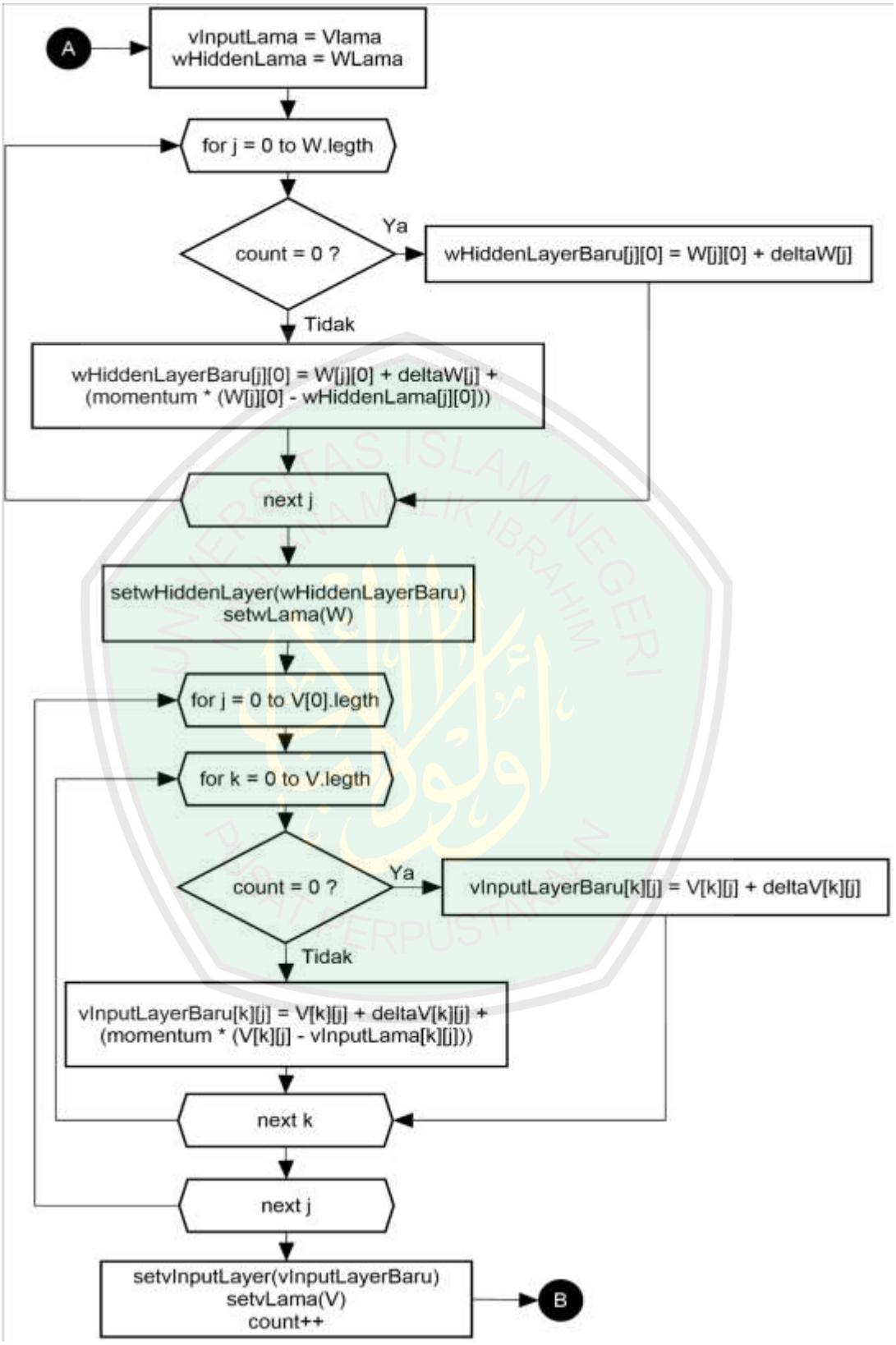
MSE=0,5 x $\{(t_{k1}-y_{k1})^2 + (t_{k2}-y_{k2})^2 + \ldots + (t_{k140}-y_{k140})^2\}$

1.5.3.2 *Training*

Proses *Training* dibuat dengan mengikuti alur yang ada dalam buku karangan (Siang,2005) yang berjudul Jaringan Syaraf Tiruan dan Pemrogramannya Menggunakan Matlab. Namun pengkodean dilakukan dengan menggunakan bahasa Java. Data yang digunakan dalam proses *training* adalah data piksel *Black and White* yang sudah ditambah dengan bias pada tiap data training pada index ke 0. Piksel-piksel itu disimpan dalam file .txt yang nanti akan dibaca oleh program penghitungan menjadi array 2 dimensi. Proses konversi *image* untuk menghasilkan piksel-piksel *Black and White* tadi dilakukan secara manual pada tiap *image* tulisan tangan. Ukuran masing-masing data training sama dengan ukuran *box area* yang ada dalam aplikasi yakni 40x40 *pixels.*

Desain dari algortima Neural Network Backpropagation sesuai dengan penggambaran alur pada buku karangan (Siang, 2005) digambarkan pada gambar 3.28.


start
Input[][]
Target [][]
Learning rate =0.7f
Momentum = 0.7f
stopCondition=0.00000000000001
maxRun = 50000
jumlahHiddenLayer = 3
jumlahOutputLayer = 1
Boolean belumTerlatih = true
count = 0
iterasi = 1
Inisilisasi Nguyen Widrow
V = V
VLama = V
W = W
WLama =W
Belum Terlatih ?
Ya
Tidak
Vterlatih[][]
Wterlatih[][]
End
MSE
MSE>stopCondition && iterasi < maxRun
Tidak
Ya
temp =0
faktorD_Out = 0
Z[0] = 1
belumTerlatih = false
iterasi++
for i= 0 to input.length
Y = 0, V=V, W=W
for k= 0 to jmlHiddenlayer
temp = 0
for l= 0 to input[0].length
temp += input[i][l] * V[l][k]
next l
Z[k + 1] = sigmoid(temp)
next k
for j = 0 to Z.length
Y += Z[j] * W[j][0]
next j
Y = sigmoid (Y)
next i
faktorD_Out = (target[i][0] - Y) * Y * (1 - Y)
for j= 0 to Z.length
deltaW[j] = learningRate * faktorD_Out * Z[j]
next j
for j= 1 to Z.length
delta_net[j] = faktorD_Out * W[j][0]
faktorD_Hidden[j] = delta_net[j] * Z[j] * (1 - Z[j])
next j
for j= 0 to jumlahHiddenLayer
for k= 0 to V.length
deltaV[k][j] = learningRate * faktorD_Hidden[j + 1] * input[i][k]
next k
next j
A
B
next i



Gambar 3. 28 Desain JST Backpropagation

Dari gambar 3.28 diketahui bahwa proses *training* dimulai dari inisialisasi bobot awal dan bobot bias dengan menggunakan metode inisialisasi dari Nguyen-Widrow. Sebelum proses berlanjut ke perambatan maju, pengecekan kondisi dilakukan untuk mengetahui apakah data sudah terlatih atau belum. Namun di awal proses kondisi "Belum Terlatih " diberi inisialisasi *true.* Maka proses berlanjut ke pengecekan MSE dan iterasi. Ketika MSE lebih dari *stop condition* dan iterasi kurang dari bilangan yang ditentukan maka proses berhenti dan bobot dianggap sudah terlatih. Jika tidak, maka proses penghitungan berlanjut ke perambatan maju. Setelah itu proses berlanjut untuk mencari suku perubahan bobot W dan V. Setelah ditemukan, maka lanjut ke proses perubahan bobot dengan memanfaatkan *momentum.* Proses perhitungan seperti ini berlanjut sampai data training terakhir. Ketika sampai pada data terakhir, maka iterasi pertama selesai. Kemudian proses berlanjut ke pengecekan MSE dan iterasi. Begitu seterusnya sampai kondisi pemberhentian proses terpenuhi.

Untuk menggambarkan algortima ini bekerja, gambar 3.29 adalah gambaran *pseudocode* dari algoritma *Backpropagation*.

```
Program Backpropagation
(Program ini dilakukan untuk pelatihan bobot)

Kamus
belumTerlatih : boolean
V[][],W[][],temp, Y, faktorD_out : double
faktorD_Hidden[], deltaW[], deltaV[][], delta_net[] : double
vInputLayerBaru[][], wHiddenLayerBaru[][] : double
vInputLama[][], wHiddenLama[][] : double
Z[], stopCondition : double
maxRun, count, iterasi : integer
```

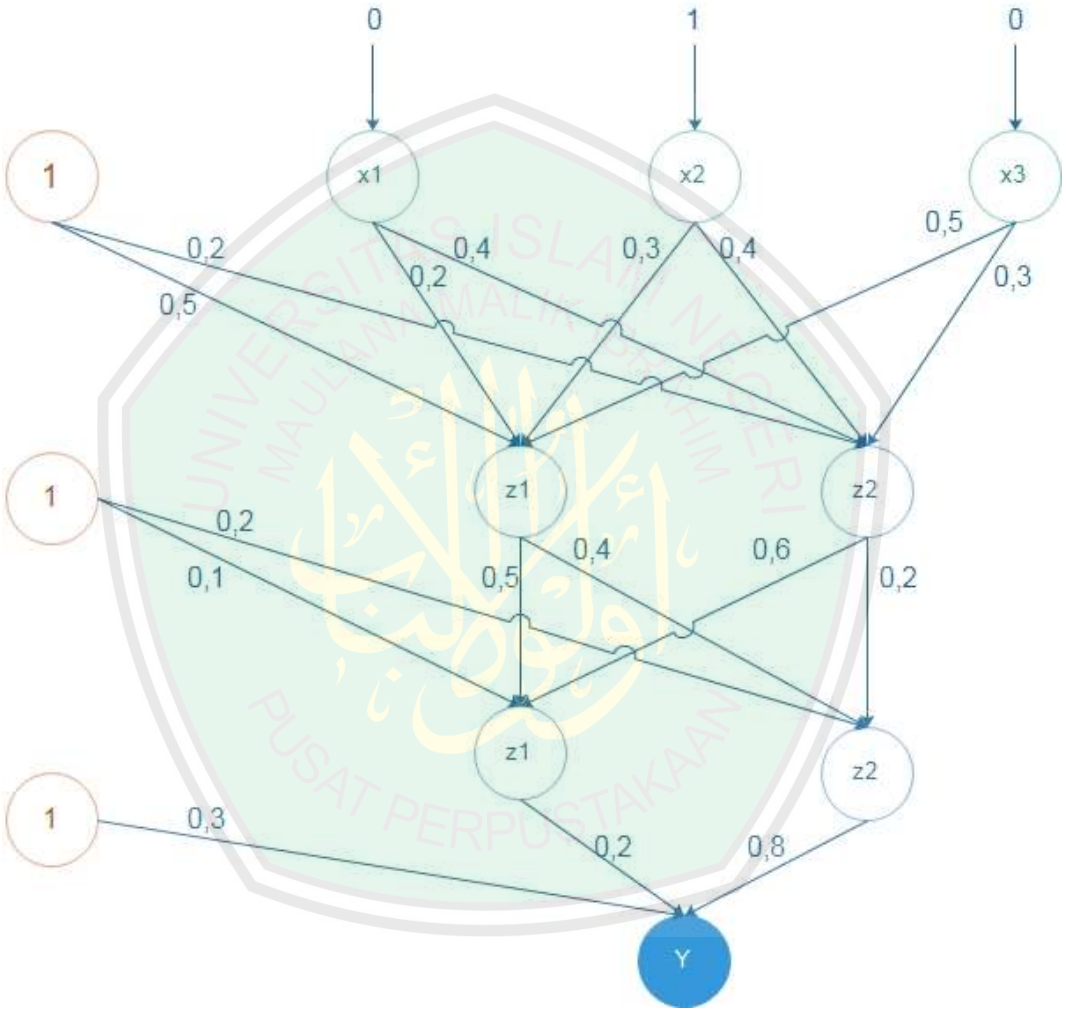
```
Kamus
Z[0] <- 1
while (belumTerlatih)
if(mse > stopCondition && iterasi < maxRnn) then
for i = 0 to panjangData do
V <- V
W <- W
Y <- 0
for k = 0 to jumlahHiddenLayer do
temp <- 0
for l = 0 to jumlahHiddenLayer do
temp +<- input[i][l] * V[l][k]
endfor
Z[k+1] <- sigmoid(temp)
endfor
for j = 0 to panjangZ do
Y +<- Z[j] * W[j][0]
endfor
Y <- sigmoid(Y)
faktorD_Out <- (target[i][0] - Y) * Y * (1 - Y)
for j = 0 to panjangZ do
deltaW[j] <- learningRate * faktorD_Out * Z[j]
endfor
for j = 1 to panjangZ do
delta_net[j] = faktorD_Out * W[j][0]
faktorD_Hidden[j] = delta_net[j] * Z[j] * (1 - Z[j])
endfor
```

```
Kamus
for j = 0 to jumlahHiddenLayer do
for k = 0 to panjangV do
deltaV[k][j] <- learningRate * faktorD_Hidden[j + 1] * input[i][k]
endfor
endfor
vInputLama <- VLama
      wHiddenLama <- WLama
for j = 0 to panjangW do
if(count <- 0) then
wHiddenLayerBaru[j][0] = W[j][0] + deltaW[j]
else
wHiddenLayerBaru[j][0] = W[j][0] + deltaW[j] + (momentum * (W[j][0]
- wHiddenLama[j][0]))
endif
endfor
W <- wHiddenLayerBaru
WLama <- WLama
for j = 0 to panjangV do
for k = 0 to panjangData do
if(count <- 0) then
vInputLayerBaru[k][j] = V[k][j] + deltaV[k][j]
else
vInputLayerBaru[k][j] = V[k][j] + deltaV[k][j] + (momentum * (V[k][j] -
vInputLama[k][j]))
endif
endfor
```

```
Kamus
endfor
V <- vInputLayerBaru
VLama <- VLama
count++
endfor
iterasi++
panggil method Backpro()
else
belumTerlatih <- false
endif
endwhile
```

Gambar 3. 29 *Pseudocode* JST Backpropagation

Dimisalkan dari pengolahan *image*, hasil untuk inputan berupa huruf dengan 3 pixel beruapa angka biner 010, menggunakan 2 *hidden layer* dan *learning rate* = 0.25, maka penggambaran dan penghitungannya diilustrasikan pada gambar 3.30 dengan penghitungan menggunakan model dari Jochen Fröhlich.



Gambar 3. 30 Ilustrasi soal JST Backpropagation

Pada tabel 3.1 ditunjukkan bahwa penghitungan dimulai dengan *feedforwardpropagation,* yakni dari penjumlahan masing-masing inputan dikalikan dengan masing-masing bobotnya. Penentuan bobot sebelumnya dilakukan secara *random.* Hasil dari perhitungan pada tahap ini adalah *output* dan *error value.*

Tabel 3. 1 Tabel perhitungan propagasi maju untuk iterasi awal

Hidden layer 1		
Input of hidden neuron 1	1*0.5 + 0 * 0.2 + 1 * 0.3 + 0 * 0.5	0.8
Input of hidden neuron 2	1*0.2+ 0 * 0.4 + 1 * 0.4 + 0 * 0.3	0.6
Output of hidden neuron 1	1/(1+EXP(-0.8))	0.689974481
Output of hidden neuron 2	1/(1+EXP(-0.6))	0.645656306
Hidden layer 2		
Input output neuron 1	1*0.1+0.689974481 * 0.5 + 0.645656306 * 0.6	0.832381024
Input hidden neuron 2	1*0.2+0.689974481 * 0.4 + 0.645656306 * 0.2	0.605121054
Output hidden neuron 1	1/(1+EXP(-0.832381024))	0.696858149
Output hidden neuron 2	1/(1+EXP(-0.605121054))	0.646827047
Output layer		
I of O layer 2	1*0.3+0.696858149 * 0.2 + 0.646827047 * 0.8	0.900441159
O of O	1/(1+EXP(-0.900441159))	0.711040152
Error value	0 - 0.711040152	-0.711040152

Setelah didapatkan *error value,* maka dilanjutkan ke langkah *backpropagation*. Pada table 3.2 dijelaskan bahwa penghitungan dimulai dengan menghitung *value* untuk merubah bobot pada matrix ke-3 atau bobot yang berada sebelum layer output.

Tabel 3. 2 Perhitungan Backpropagation dan pembaruan bobot pada matrix ke-3

Weight matrix 3		
Value for changing weight 1	0.25 * (-0.711040152) * 0.696858149 * 0.711040152 *(1-0.711040152)	-0.02545131
Value for changing weight 2	0.25 * (-0.711040152) * 0.646827047 * 0.711040152 *(1-0.711040152)	-0.023624027
Value for changing weight Bias	0.25 * (-0.711040152) * 1 * 0.711040152 *(1-0.711040152)	-0.036522943
change weight 1	0.2 + (-0.02545131)	0.22423372
change weight 2	0.8 + -0.023624027)	0.82165945
change weight Bias	0.3 + (-0.036522943)	0.82165945

Setelah semua bobot pada matrix ke-3 berubah, proses dilanjutkan dengan pembaruan bobot pada matrix ke-2 atau bobot yang bereada sebelum hidden layer yang ke-2. Tabel 3.3 menggambarkan proses perubahan bobot pada matrix ke-2.

Tabel 3. 3 Perhitungan pembaruan bobot pada matrix ke-2

Weight matrix 2		
Value for changing weight 1	0.25 * (-0.711040152) * 0.689974481 * 0.696858149 *(1-0.696858149)	-0.025909405
Value for changing weight 2	0.25 * (-0.711040152) * 0.689974481 * 0.646827047 *(1-0.646827047)	-0.028018364
Value for changing weight 3	0.25 * (-0.711040152) * 0.645656306* 0.696858149 *(1-0.696858149)	-0.024245202
Value for changing weight 4	0.25 * (-0.711040152) * 0.645656306 * 0.646827047 *(1-0.646827047)	-0.026218699
Value for changing weight Bias 1	0.25 * (-0.711040152) * 1* 0.696858149 *(1- 0.696858149)	-0.037551251
Value for changing weight Bias 2	0.25 * (-0.711040152) * 1* 0.646827047 *(1-0.646827047)	-0.040607826
change weight 1	0.5 +(-0.025909405)	0.474090595
change weight 2	0.4 +(-0.028018364)	0.371981636
change weight 3	0.6 +(-0.024245202)	0.575754798
change weight 4	0.2 + (-0.026218699)	0.173781301
change weight Bias 1	0.1 + (-0.037551251)	0.062448749
change weight Bias 2	0.2 + (-0.040607826)	0.159392174

Setelah semua bobot pada matrix ke-2 berubah, proses dilanjutkan dengan pembaruan bobot pada matrix ke-2 atau bobot yang bereada sebelum hidden layer yang ke-1, dan pelatihan untuk iterasi pertama data pertama selesai. Tabel 3.4 menggambarkan proses perubahan bobot pada matrix ke-1.

Tabel 3. 4 Perhitungan pembaruan bobot pada matrix ke-1

Weight matriks 1		
Value for changing weight 1	0.25 * (-0.711040152) * 0 * 0.689974481 *(1- 0.689974481)	0
Value for changing weight 2	0.25 * (-0.711040152) * 0 * 0.645656306 *(1- 0.645656306)	0
Value for changing weight 3	0.25 * (-0.711040152) * 1 * 0.689974481 *(1- 0.689974481)	-0.038024596
Value for changing weight 4	0.25 * (-0.711040152) * 1 * 0.645656306 *(1- 0.645656306)	-0.040668695
Value for changing weight 5	0.25 * (-0.711040152) * 0 * 0.689974481 *(1- 0.689974481)	0
Value for changing weight 6	0.25 * (-0.711040152) * 0 * 0.645656306 *(1- 0.645656306)	0
Value for changing weight Bias 1	0.25 * (-0.711040152) * 1 * 0.689974481 *(1- 0.689974481)	-0.038024596
Value for changing weight Bias 2	0.25 * (-0.711040152) * 1 * 0.645656306 *(1- 0.645656306)	-0.040668695
change weight 1	0.2+0	0.2
change weight 2	0.4+0	0.4
change weight 3	0.3+(-0.038024596)	0.261975404
change weight 4	0.4+(-0.040668695)	0.359331305
change weight 5	0.5+0	0.5
change weight 6	0.3+0	0.3
change weight Bias 1	0.5 + (-0.038024596)	0.461975404
change weight Bias 2	0.2 + (-0.040668695)	0.159331305

Iterasi pertama sudah selesai dilakukan dengan menghasilan bobot-bobot yang baru untuk iterasi selanjutnya. Prosedur yang sama digunakan untuk pola inputan selanjutnya, tetapi dengan bobot yang berbeda. Setelah merambat maju dan mundur pada pola kedua, satu kali *step* pembelajaran sudah selesai dan nilai error

bias dihitung. Dengan melakukan prosedur ini berulang-ulang, maka nilai error akan semakin kecil. Algoritma ini dianggap sudah berhasil apabial nilai error mendekati 0, dan ini merupakan nilai sempurna atau bias juga mendekati nol.

Menurut (Siang, 2005), tujuan utama penggunaan *Backpropagation* adalah mendapatkan keseimbangan antara pengenalan pola pelatihan secara benar dan respon yang baik untuk pola lain yang sejenis (disebut pengujian). Jaringan dapat dilatih terus menerus hingga semua pola pelatihan dikenali dengan benar. Akan tetapi hal itu tidak menjamin jaringan akan mengenali pola pengujian dengan tepat. Jadi tidaklah bermanfaat untuk meneruskan iterasi hingga semua kesalahan pola pelatihan = 0.

1.5.3.3 *Testing*

Proses berikutnya adalah proses penghitungan pengujian (*testing*). Proses ini dilakukan di dalam system aplikasi android setelah beberapa *preprocessing* dilakukan pada pixel yang diterima kamera. Contoh hasil perhitungan *Feedforwardpropagation* dari bobot yang sudah dilatih dalam *Backpropagation* dengan data pelatihan sebanyak 4 data training, dan target yang diharapkan adalah {0.23}, {0.24}, {0.25}, {0.26} digambarkan pada gambar 3.31.

```
perubahan bobot tersembunyi
-0.909872659952393, -0.8024296312669948, -0.09689032551114606,
0.4323119645638562, -0.70011319093848, 0.6383248237906571,
1.056535742133928, -1.0454721769948947, -0.9726851937314365,

perubahan bobot keluaran
-1.3278235192785564
0.3085929548248517
-0.225266943325002
0.2245498789969228
```

Gambar 3. 31 Contoh hasil training

Dari contoh hasil training di atas, maka bisa dilakukan pengujian dengan pola inputan yang dijadikan training. Pola yang diinputkan untuk percobaan training digambarkan pada gambar 3.32.

```
Int input[][] ={ {1,0,0},
                 {1,0,1},
                 {1,1,0},
                 {1,1,1},};
```

Gambar 3. 32 Contoh pola inputan untuk training

Berikut adalah perhitungan *testing* untuk data pertama.

Z1 = (1 * -0.909872659952393) + (0* 0.432311964563856) + (0* 1.056535742133928)

Z2 = (1 * -0.8024296312669948) + (0*-0.70011319093848) +(0* -1.0454721769948947)

Z3 = (1 * -0.096890325511146) +(0*0.6383248237906571) +(0* -0.9726851937314365)

Perhitungan sigmoid

$z1 = \frac{1}{1+ e^{-(-0.909)}} = 0.287026$

$z1 = \frac{1}{1+ e^{-(-0.802)}} = 0.309506$

$z1 = \frac{1}{1+ e^{-(-0.097)}} = 0.475796$

Perhitungan hasil

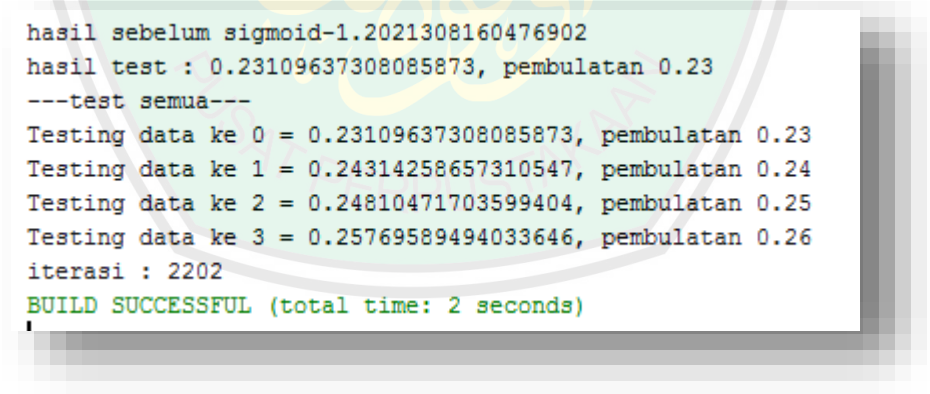
Y = (1*-1.3278235192785564) + (0.287026* 0.3085929548248517) +

(0.309506 * -0.225266943325002) + (0.475796 * 0.2245498789969228)

Y = -1.20213

$Y = \frac{1}{1+ e^{-(-1.20213)}} = 0.231096 = 0.23$

Dari percobaan pengujian dihasilkan nilai yang sesuai dengan target yang diharapkan setelah proses pembulatan. Hasil *testing* yang dilakukan di Netbeans digambarkan pada gambar 3.33.



Gambar 3. 33 Contoh hasil testing pada semua data

BAB IV

HASIL DAN PEMBAHASAN

Pada bab ini dijelaskan mengenai hasil dari pembuatan aplikasi dan hasil uji terhadap *marker-marker* yang tersedia dengan device android yang berbeda. Uji coba ditujukan untuk melihat sejauh mana keberhasilan dari implementasi perangkat lunak ini. Selanjutnya akan dilaksanakan evaluasi dengan melakukan analisa terhadap hasil uji coba dan juga untuk mendapatkan kesimpulan dan saran untuk pengembangan ke depan bagi implementasi aplikasi ini.

4.1 Hasil Implementasi

Hasil implementasi dari rancangan yang sudah dibuat dimulai dari halaman Splashscreen yang muncul selama 5 detik. Setelah itu langsung menuju halaman utama yakni halaman proses *Augmented Reality*. Proses pengenalan huruf dilakukan dengan menempatkan huruf yang di tes pada *box area.* Setelah itu masih harus mencari area yang benar-benar pas dari huruf agar *pixel* yang dikenali bisa sesuai dengan huruf-huruf ketika di training.

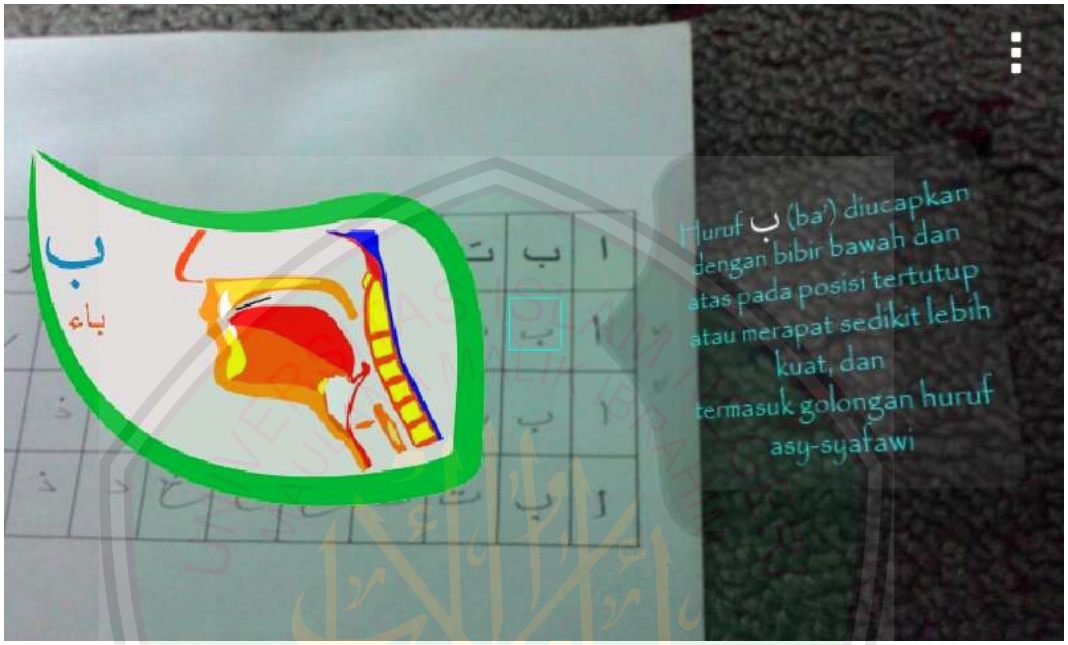
Beberapa penampakan dari proses aplikasi berjalan dari *Splash Screen* sampai pada tombol-tombol yang ada pada aplikasi dijelaskan dan digambarkan sebagai berikut: *Pertama*, yaitu penampakan hasil impelementasi pada antarmuka *Splesh Screen.* Antarmuka ini muncul selama 5 detik sebelum masuk ke halaman utama, dan berisi logo serta informasi tentang pembuat.

Gambar 4. 1 Penampakan Splash Screen

Setelah halaman *Splash Screen* selesai, kemudian masuk ke halaman utama, yakni halaman untuk memproses pixel yang masuk dengan *box area* diatasnya.

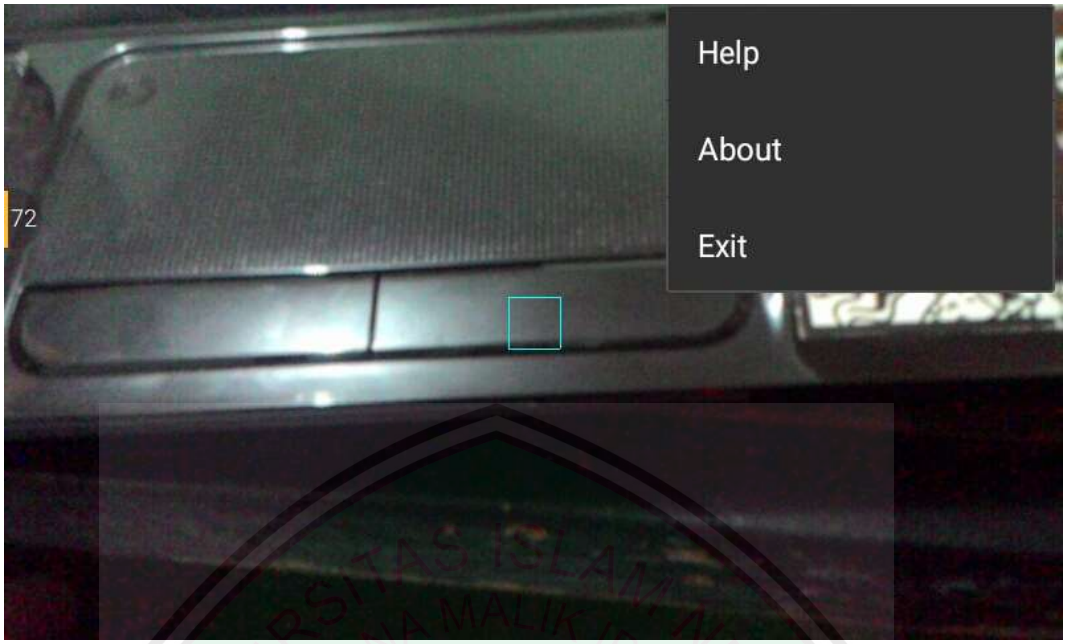
Gambar 4. 2 Halaman Camera View AR

Selanjutnya, setelah target yang sesuai dari proses perhitungan dari *pixel* ditemukan, maka akan ditampilkan animasi tentang pengucapan huruf dan keterangan dari huruf.



Gambar 4. 3 Contoh hasil pemunculan animasi dan keterangan huruf

Yang terakhir untuk penampakan antarmuka menu. Menu yang tersedia ada 3 tombol, yakni tombol *Help*, tombol *About*, dan tombol *Exit*.



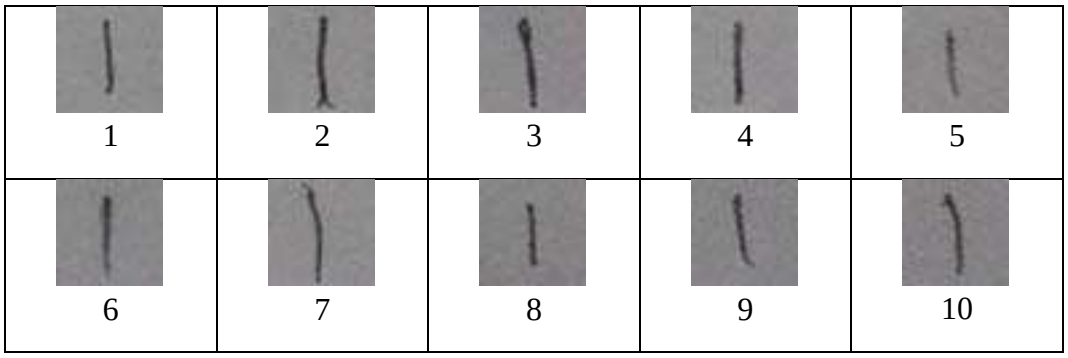
Gambar 4. 4 Penampakan Menu

4.2 Hasil Uji Coba

Marker-marker yang dipakai untuk menguji aplikasi yakni:

a. Alif

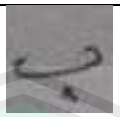
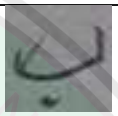
Huruf Alif yang digambarkan pada gambar 4.5 berjumlah 10 gambar, dan berukuran 40x40 *pixels.* Dari beberapa contoh marker alif ini dipilih gambar nomor 1, 2, 3, dan 4 untuk data pengujian.



Gambar 4. 5 Kumpulan marker Alif

b. Ba

Huruf Ba yang digambarkan pada gambar 4.6 berjumlah 10 gambar, dan berukuran 40x40 *pixels*. Dari beberapa contoh marker ba ini dipilih gambar nomor 1, 2, 3, dan 4 untuk data pengujian.

 1	 2	 3	 4	 5
 6	 7	 8	 9	 10

Gambar 4. 6 Kumpulan marker Ba

c. Ta

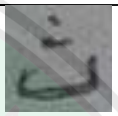
Huruf Ta yang digambarkan pada gambar 4.7 berjumlah 10 gambar, dan berukuran 40x40 *pixels*. Dari beberapa contoh marker ta ini dipilih gambar nomor 1, 2, 3, dan 4 untuk data pengujian.

 1	 2	 3	 4	 5
 6	 7	 8	 9	 10

Gambar 4. 7 Kumpulan marker Ta

d. Tsa

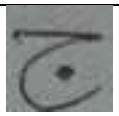
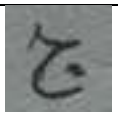
Huruf Tsa yang digambarkan pada gambar 4.8 berjumlah 10 gambar, dan berukuran 40x40 *pixels.* Dari beberapa contoh marker tsa ini dipilih gambar nomor 1, 2, 3, dan 4 untuk data pengujian.

 1	 2	 3	 4	 5
 6	 7	 8	 9	 10

Gambar 4. 8 Kumpulan marker Tsa

e. Jim

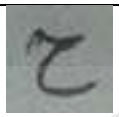
Huruf Jim yang digambarkan pada gambar 4.9 berjumlah 10 gambar, dan berukuran 40x40 *pixels.* Dari beberapa contoh marker jim ini dipilih gambar nomor 1, 2, 3, dan 4 untuk data pengujian.

 1	 2	 3	 4	 5
 6	 7	 8	 9	 10

Gambar 4. 9 Kumpulan marker Jim

f. Kha

Huruf Kha yang digambarkan pada gambar 4.10 berjumlah 10 gambar, dan berukuran 40x40 *pixels.* Dari beberapa contoh marker kha ini dipilih gambar nomor 1, 2, 3, dan 4 untuk data pengujian.

 1	 2	 3	 4	 5
 6	 7	 8	 9	 10

Gambar 4. 10 Kumpulan marker Kha

g. Kho

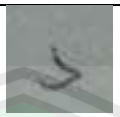
Huruf Kho yang digambarkan pada gambar 4.11berjumlah 10 gambar, dan berukuran 40x40 *pixels.* Dari beberapa contoh marker kho ini dipilih gambar nomor 1, 2, 3, dan 4 untuk data pengujian.

 1	 2	 3	 4	 5
 6	 7	 8	 9	 10

Gambar 4. 11 Kumpulan marker Kho

h. Dal

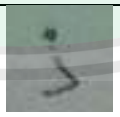
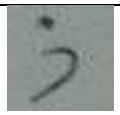
Huruf Dal yang digambarkan pada gambar 4.12 berjumlah 10 gambar, dan berukuran 40x40 *pixels.* Dari beberapa contoh marker dal ini dipilih gambar nomor 1, 2, 4, dan 5 untuk data pengujian.

 1	 2	 3	 4	 5
 6	 7	 8	 9	 10

Gambar 4. 12 Kumpulan marker Dal

i. Dzal

Huruf alif yang digambarkan pada gambar 4.13 berjumlah 10 gambar, dan berukuran 40x40 *pixels.* Dari beberapa contoh marker dzal ini dipilih gambar nomor 1, 2, 3, dan 4 untuk data pengujian.

 1	 2	 3	 4	 5
 6	 7	 8	 9	 10

Gambar 4. 13 Kumpulan marker Dzal

j. Ro

Huruf Ro yang digambarkan pada gambar 4.14 berjumlah 10 gambar, dan berukuran 40x40 *pixels.* Dari beberapa contoh marker ro ini dipilih gambar nomor 1, 2, 3, dan 4 untuk data pengujian.

 1	 2	 3	 4	 5
 6	 7	 8	 9	 10

Gambar 4. 14 Kumpulan marker Ro

Pendeteksian marker dengan komparasi *feedforwardpropagation* dengan marker disesuaikan dengan *box area* dengan ukuran 40x40 *pixels* sebagai batasan pengambilan marker. Pendeteksian mungkin akan memakan waktu beberapa lama karena inputan yang sesuai dengan marker tidak bisa langsung ditemukan. Dari data yang ada di atas pengujian dilakukan dengan mengambil 4 sampel dari masing-masing huruf sehingga jumlah total untuk pengujian adalah 40 huruf hijaiyah. Pengujian ini menggunakan 1 device android yakni, Zenfone 4 . Hasil pengujian dijelaskan dalam table 4.1:

Tabel 4. 1 Pengujian pada device Android Zenfone 4 Lolipop

No.	Data	Isi	Hasil	Keterangan
1	Alif 1	ا	ا	Sesuai
2	Alif 2	ا	ا	Sesuai
3	Alif 3	ا	ا	Sesuai
4	Alif 4	ا	ا	Sesuai
5	Ba 1	ب	ب	Sesuai
6	Ba 2	ب	ب	Sesuai
7	Ba 3	ب	ب	Sesuai
8	Ba 4	ب	ب	Sesuai
9	Ta 1	ت	ت	Sesuai
10	Ta 2	ت	ت	Sesuai
11	Ta 3	ت	ت	Sesuai
12	Ta 4	ت	ت	Sesuai
13	Tsa 1	ث	ث	Sesuai
14	Tsa 2	ث	ث	Sesuai
15	Tsa 3	ث	ث	Sesuai
16	Tsa 4	ث	ث	Sesuai
17	Jim 1	ج	ج	Sesuai
18	Jim 2	ج	د	Tidak Sesuai
19	Jim 3	ج	د	Tidak Sesuai
20	Jim 4	ج	ج	Sesuai
21	Kha 1	ح	ح	Sesuai
22	Kha 2	ح	خ	Tidak Sesuai
23	Kha 3	ح	ج	Tidak Sesuai
24	Kha 4	ح	خ	Tidak Sesuai
25	Kho 1	خ	خ	Sesuai
26	Kho 2	خ	ج	Tidak Sesuai
27	Kho 3	خ	ث	Tidak Sesuai
28	Kho 4	خ	ث	Tidak Sesuai
29	Dal 1	د	ج	Tidak Sesuai
30	Dal 2	د	ح	Tidak Sesuai
31	Dal 3	د	د	Sesuai
32	Dal 4	د	د	Sesuai
33	Dzal 1	ذ	د	Tidak Sesuai
34	Dzal 2	ذ	د	Tidak Sesuai
35	Dzal 4	ذ	ذ	Sesuai
36	Dzal 5	ذ	ذ	Sesuai
37	Ro’ 1	ر	ر	Sesuai
38	Ro’ 2	ر	د	Tidak Sesuai
39	Ro’ 3	ر	ر	Sesuai
40	Ro’ 4	ر	ر	Sesuai
Presentase keberhasilan				67,5 %

4.3 Pembahasan

Ketika dilakukan pengujian, data tidak langsung ditemukan. Akan tetapi sambil digerak-gerakan sesuai batas area kotak (*box area*). Uji coba dilakukan dengan posisi marker tegak lurus dengan kamera. Jarak kamera disesuaikan dengan lebar dari *image* yang di tes dimana image bisa terlihat keseluruhan pada *box area*. Hasil akurasinya masih buruk karena jarak sangat menentukan. Nilai *pixel* inputan dari *camera view* selalu berubah-ubah dan tidak stabil, maka dari itu untuk menemukan nilai yang cocok membutuhkan waktu. Dari segi kenyamanan, hal ini pasti sangat berpengaruh.

Dari hasil coba yang dilakukan, diketahui bahwa prosentase keberhasilan masih maksimal 67,5 % dan data yang ditargetkan tidak bisa langsung muncul, hal ini terjadi karena hasil dari *preprocessing* ketika *training* dan *testing* yang berupa masukan nilai *Black and White* tidak selalu sama.

Faktor cahaya dan cara masuknya *pixel* sangat menentukan. Maka uji coba dilakukan di tempat yang cahayanya merata. Masalah yang ditimbulkan oleh cahaya yang tidak merata tersebut mengakibatkan sistem mengambil nilai tengah yang bisa jadi tidak sama dengan nilai ketika proses *training* dilakukan. Ketika nilai *Threshold* tidak sama, secara otomatis nilai yang di dapat untuk pixel putih dan hitam juga tidak sama. Maka masukan yang didapat akan susah dikenali oleh sistem

4.4 Augmented Reality Untuk Pembelajaran Makhorijul Huruf

Konsep pembelajaran atau pencarian ilmu apa saja termasuk cara membaca Al-Quran tidak mesti sama. Beberapa lembaga pendidikan Al-Quran pasti memiliki cara-cara tersendiri dalam penyampaiannya. Termasuk dalam penelitian tentang *Augmented Reality* ini yang dikhususkan untuk pembelajaran makhorijul huruf dengan memanfaatkan teknologi. Walaupun memang belum cukup apabila kita belajar Al-Quran hanya memanfaatkan teknologi. Dalam buku "*100 Tanya Jawab Al-Quran*", kumpulan dari jawaban pertanyaan yang diajukan kepada Ustadz Maftuh Basthul Birri, Pengasuh Madrasatul Murottilil Quran, Pondok Pesantren Lirboyo, Kediri tentang masalah Al-Quran. Dijelaskan bahwa belajar Al-Quran dengan alat penyampai apapun termasuk teknologi android itu bisa dan boleh saja dilakukan. Akan tetapi, semua itu tetap belum cukup kalau tanpa dengan guru langsung yang ahli. Karena alat penyampai itu tidak bisa menegur, menyalahkan dan membetulkan bacaan. Maka dari itu, teknologi sebagai alat penyampai pembelajaran Al-Quran itu hanya bersifat penunjang dan penarik minat dalam proses pembelajarannya.

Penunjang-penunjang seperti teknologi *Augmented Reality* ini diharapkan dapat menunjang cara pengajaran Al-Quran agar siswa menjadi faham dan benar cara bacanya. Ini semua bertujuan agar kita bisa memuliakan Al-Quran salah satunya dengan membacanya dengan benar sesuai dengan ilmu tajwid karena memang Al-Quran sendiri itu sudah mulia diturunkan pada malam yang mulia.

إِنَّآ أَنزَلْنَٰهُ فِى لَيْلَةِ ٱلْقَدْرِ ﴿١﴾

"Sesungguhnya Kami telah menurunkannya (Al Quran) pada malam kemuliaan"

Selanjutnya juga diharapkan dengan bacaan-bacaan yang fasih itu bisa menambah iman kita yang membaca juga yang mendengar. Karena bisa jadi seseorang yang mendengarkan bacaan Al-Quran sesesorang yang fasih bacaannya serta enak lagunya bisa membuat hatinya luluh. Firman Allah dalam Q.S Al-Anfal ayat 2 :

إِنَّمَا ٱلْمُؤْمِنُونَ ٱلَّذِينَ إِذَا ذُكِرَ ٱللَّهُ وَجِلَتْ قُلُوبُهُمْ وَإِذَا تُلِيَتْ عَلَيْهِمْ

ءَايَٰتُهُۥ زَادَتْهُمْ إِيمَٰنًا وَعَلَىٰ رَبِّهِمْ يَتَوَكَّلُونَ ۝٢

"Sesungguhnya orang-orang yang beriman ialah mereka yang bila disebut nama Allah gemetarlah hati mereka, dan apabila dibacakan ayat-ayat-Nya bertambahlah iman mereka (karenanya), dan hanya kepada Tuhanlah mereka bertawakkal"

Selain mulia, Al-Quran juga menjadi sebab diangkatnya derajat seseorang. Apalagi kalau bacaannya fasih. Dalam H.R Muslim dijelaskan bahwa:

ان الله تعالى يرفع بهذا الكتاب اقواما ويضع به اخرين

(رواه مسلم عن عمر- التبيان في الباب الاول)

" Sesungguhnya Allah Swt mengangkat derajat kaum karena Al-Quran ini dan merendahkan pada kaum yang lain karena Al-Quran."

Selain itu, Al-Quran juga bisa melaknat orang-orang yang membacanya. Dalam kitab Nihayatul Qoulil Mufid diterangkan bahwa ada tiga orang yang rentan dilaknat oleh Al-Quran :

1. Orang yang bacaannya betul-betul menyimpang jauh dari ketentuan tajwid dan sampai merubah lafadz-lafadznya.
2. Orang yang bacaannya bisa merusak dan merubah ma'na
3. Orang yang membaca Al-Quran namun pembuat sehari-harinya sangat jauh dari nilai-nilai Al-Quran.

Maka dari itu, aplikasi ini diharapkan bisa menunjang cara pembelajaran Al-Quran khususnya pada segi makhorijul hurufnya. Hal ini dimaksudkan agar kita bisa memuliakan Al-Quran, mendapat kemuliaan melalui Al-Quran, dan juga bisa terhindar dari laknat Al-Quran.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil implementasi dan uji coba yang dilakukan dapat disimpulkan bahwa:

a. Algoritma *Neural Network Backpropagation* sebenarnya sesuai digunakan untuk penelitian penganalan *image*. Namun setelah testing dilakukan, prosesnya sedikit berat untuk penelitian ini.
b. Masukan dari *camera view* masih belum stabil. Sehingga *pixel* yang dimasukan tidak selalu sama dan kemunculan target bisa memakan waktu dan juga bisa salah dalam memilih target.
c. Hasil uji coba dari Algoritma ini untuk mengenali tulisan tangan huruf hijaiyah adalah 67,5 %, dengan posisi kamera menyesuaikan ukuran gambar.

5.2 Saran

Beberapa saran ntuk penelitian dan pengembangan aplikasi selanjutnya adalah sebabagi berikut :

a. Dapat digunakan metode lain untuk pengambilan pixel dari *camera view*. Sebab nilai *pixel* yang selalu berubah menjadikan sistem kurang bisa mengenali target atau *marker.*
b. Jika tetap meneruskan memakai JST *Backpropagation*, maka perlu ditambah varisasi lain sehingga proses *traininig* bisa lebih cepat dan akurat. Karena dalam

penelitian ini hanya dilakukan penambahan momentum dan inisilaisasi bobot dari Nguyen Widrow.

Metode *thresholding* bisa menggunakan metode lainnya karena dari beberapa *review* diketahui bahwa cahaya yang kurang merata sangat berperngaruh dalam metode ini.

DAFTAR PUSTAKA

Abdullah. 1994. *Lubaabut Tafsiir Min Ibni Katsiir.* Jakarta : PUSTAKA IMAM ASY-SYAFI'I

Alihla, B., Kwaik, K. 2012 *OIAHCR: Online Isolated Arabic handwritten Character Recognition Using Neural Network.* Islamic University of Gaza Palestine.

Alquran Al-Karim

Bahresiy, S. 1992. *Terjemahan Singkat Tafsir Ibnu Katsir Jilid 8.* Surabaya: PT Bina Ilmu Offset

Balint, Z., Magyari, B., Simon, K. 2012. *Augmented Reality and Image Recognition Based Framework for Treasure Hunt Games.* Babes-Bolyai University.

Birri, MB. 2010. *100 Tanya Jawab Al-Quran.* Lirboyo : MMQ Lirboyo

Dimas. 2011. *Android : Retrieving the Camera preview as a Pixel Array*.(online) (http://www.41post.com/3470/programming/android-retrieving-the-camera-preview-as-a-pixel-array) diakses tanggal 13 juni 2015 pukul 12.00

Frohlich, J. 2015. *Nerual Network with JAVA 2004 Edition:Backpropagation.* (online) (http://www.nnwj.de/backpropagation.html), diakses tanggal 28 Februari pukul 21.00

Hermanto, Purnawan, WR. 2009. *Prediksi Produksi Kelapa Sawit Berdasarkan Kualitas Lahan Menggunakan Model Artificial Neural Network(ANN).* INSTIPER Yogyakarta

Hermawan, A. 2006. *Jaringan Saraf Tiruan : Teori dan Aplikasi.* Yogyakarta: Penerbit ANDI

Kementrian Agama RI. 2010. *Al-Quran dan Tafsirnya.* Jakarta : Penerbit Lentera Abadi

Ketai. (online) (http:// ketai.prg) diakses tanggal 2 Oktober 2015 pukul 00.25

Krevelen, V., Poelman, R. 2010. *A Survey of Augmented Reality Technologies, Applications and Limitations.* Netherlands : Delfi University of Technology

Liu, Z., Cai, J., Buse, R. *Handwriting Recognition: Soft Computing and Probabilistic Approaces.* 2003. New York : Springer –Verlag Berlin Heidelberg.

Long, J. 2010. *DroidAR : DroidAR Mobile Locationbased Augmented Reality Framework for Android. (*online*) (*http://bitstars.github.io/droidar/*),* diakses pada tanggal 5 Maret 2015 pukul 22.25

Nurmala, N., Sugiharto, A., Sarwoko, EA.2010. *Algoritma Backpropagation Untuk Pengenalan Pola Karakter Huruf Jawa.* UNDIP

Permana, MI., Kuswardayan, I., Hariadi, RR. 2013. *Implementasi Penanda Bebas pada Game Tower Defense Berbasis Augmented Reality.* Institut Teknologi Sepuluh November

Purnomo, HM., Kurniawan, A. 2006. *Supervised Neural Network dan aplikasinya.* Yogyakarta: Penerbit Graha Ilmu

Raffety, D. 2010. Android Camera Preview is Sideways . (online) (http://stackoverflow.com/questions/3841122/android-camera-preview-is-sideways) diakses tanggal 2 Oktober 2015 pukul 00.20

Siang, JJ. 2005. *Jaringan Syaraf Tiruan dan Pemrogramannya Menggunakan Matlab.* Yogyakarta: Penerbit ANDI

Sinambela, DP., Sitorus, SH. 2013. *Pengenalan Karakter Tulisan Tangan Latin pada Jaringan Saraf Tiruan Metode Backpropagatin dengan Input Citra Kamera Digital.* Universitas Mpu Tantular Jakarta, Universitas Tanjungpura Pontianak.

Sood, R. 2012. *Pro Augmented Reality*. New York : Apress.

Verma, R., Kaur, R. 2014, *An Efficient Technique for Character Recognition Using Neural Network & Feature Extraction.* Mandi Gobindgarh Punjab, India.

Vicky. 2012. Pengertian pemrograman Java-Kelebihan dan Kekurangan. (online) http://belajar-komputer-mu.com/pengertian-pemrograman-java-kelebihan-dan-kekurangan/ , diakses tanggal 4 Juni 2015 pukul 20.25

Yudi. 2014. *Apa itu OpenGL dan OpenGL-ES.* (online) (http://indonesiaberkicau.com/apa-itu-opengl/) diakses tanggal 29 September 2015 pukul 08.15

Yunus, M. 1985. *Tafsir Quran Karim.* Jakarta: PT. HIDAKARYA AGUNG

Zechner, M. 2013. *Goals and Features.* (online) (http://libgdx.badlogicgames.com/features.html), diakses tanggal 5 Maret 2015 pukul 21.00

http://belajarmembacaalquran.com/membaca-al-quran-dengan-tajwid/diakses tanggal 12 juni 2015 pukul 18.30

http://bitstars.com/other-projects/droidar-ar-framework/ diakses tanggal 2 Oktober 2015 pukul 00.17