TUGAS AKHIR – TE141599

PENERJEMAHAN BAHASA ISYARAT INDONESIA MENGGGUNAKAN KAMERA PADA TELEPON GENGGAM ANDROID

Muhammad Yunus A.
NRP 2212100114

Dosen Pembimbing
Dr. Ir. Djoko Purwanto, M.Eng.
Ronny Mardiyanto, S.T., M.T., Ph.D.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT – TE141599

REAL TIME INDONESIAN SIGN LANGUAGE RECOGNITION USING CAMERA IN ANDROID MOBILE PHONE

Muhammad Yunus A.
NRP 2212100114

Supervisor
Dr. Ir. Djoko Purwanto, M.Eng.
Ronny Mardiyanto, S.T., M.T., Ph.D.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty of Industrial Technology
Sepuluh Nopember Institute of Technology
Surabaya 2016

PENERJEMAHAN BAHASA ISYARAT INDONESIA MENGGGUNAKAN KAMERA PADA TELEPON GENGGAM ANDROID

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Elektronika
Jurusan Teknik Elektro
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember

Menyetujui

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Ir. Djoko Purwanto, M.Eng.	Ronny Mardiyanto, ST., MT., Ph.D.
NIP. 196512111990021002	NIP. 198101182003121003

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

SURABAYA
JUNI, 2016

PENERJEMAHAN BAHASA ISYARAT INDONESIA MENGGGUNAKAN KAMERA PADA TELEPON GENGGAM ANDROID

Nama : **Muhammad Yunus A.**
Pembimbing I : **Dr. Ir. Djoko Purwanto, M.Eng.**
Pembimbing II : **Ronny Mardiyanto, S.T., M.T., Ph.D.**

ABSTRAK

Penginderaan visual atau *machine vision* merupakan suatu proses manipulasi data citra. Data tersebut dapat digunakan untuk melakukan intepretasi banyak hal, salah satunya yaitu pengenalan *gesture* dengan metode segmentasi. Pengenalan gesture dengan metode segmentasi adalah antarmuka yang dapat mengenali gerak-isyarat seorang manusia dan mentranslasikan gerakan tersebut sebagai instruksi yang dapat dipahami oleh komputer. Pengenalan *gesture* dapat digunakan untuk penerjemahkan bahasa isyarat pada orang tunawicara. Hal ini karena banyaknya orang yang tidak mengerti bahasa tangan orang tunawicara. Sehingga, orang tunawicara kesulitan dalam berinteraksi di masyarakat.

Pada tugas akhir ini pengenalan gesture untuk penerjemahan bahasa isyarat lebih mengarah pada *hand recognition*, yaitu pendeteksian perubahan gerak tangan, dengan menggunakan android mobile phone sebagai divaisnya. *Android mobile phone* memiliki kamera untuk menangkap citra orang tuna wicara saat berkomunikasi menggunakan bahasa isyarat berupa gerakan tangan. Selanjutnya, citra diproses oleh *processing unit android* untuk melakukan proses *hand recognition*. Setelah proses tersebut selesai, maka layar *display* akan memunculkan huruf atau kata dari perubahan posisi gerak tangan yang dilakukan orang tunawicara yang berada di depan kamera.

Hasil pengujian yang dilakukan pada tugas akhir ini adalah sistem dapat mengenali gesture dari pergerakan tangan pengguna dan juga dapat memprediksi label dengan akurat. Dalam kondisi pencahayaan yang cukup, Akurasi tertinggi dalam penentuan posisi target di 50 cm, yaitu dengan kesalahan pengukuran bernilai kurang dari 2%.

Kata kunci: pengenalan *gesture*, kamera, *machine vision, android*, *hand recognition*.

REAL TIME INDONESIAN SIGN LANGUANGE RECOGNITION USING CAMERA IN ANDROID MOBILE PHONE

Name	**: Muhammad Yunus A.**
Supervisor	**: Dr. Ir. Djoko Purwanto, M.Eng.**
Co-Supervisor	**: Ronny Mardiyanto, S.T., M.T., Ph.D.**

ABSTRACT

Machine vision is a process of manipulation of image data. Such data can be used to interpret many things, one of which is the introduction of gesture using segmentation color. Gesture recognition with segmentation is an interface that can recognize a human gestures and translates that movement as an instruction that can be understood by computers. Gesture recognition can be used for translating sign language on the mute people. This is because many people do not understand the language of the hands of the mute people. Thus, the mute people difficulty interacting in the community.

In this experiment, gesture recognition for sign language translation is more directed to hand recognition, namely the detection of changes in gestures, using android mobile phone as its devices. Android mobile phone has a camera to capture an image of the current speech impaired to communicate using sign language in the form of hand gestures. Furthermore, the image is processed by the processing unit android to make the process of hand recognition. Once the process is completed, then the display screen will bring up a letter or a word of the change of position of hand gestures made speech impaired people who are in front of the camera.

The results of testing on this final task is the system can recognize the gesture of the hand movements of the user and can also predict the label accurately. In the lighting conditions, the highest accuracy in the determination of the position of the target at 50 cm, with an error of measurement of value is less than 2%.

Keywords: gesture recognition, camera, machine vision, android, hand recognition.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1. Latar Belakang

Berkomunikasi merupakan kebutuhan manusia dalam berinteraksi antara satu dengan yang lain. Banyak cara yang dilakukan unutk berkomunikasi,diantaranya adalah dengan berbicara melalui lisan atau dengan tangan melalui bahasa isyarat, serta tulisan. Dalam masyarakat terdapat kaum tunawicara yang karena keterbasannya tidak dapat menggunakan bahasa lisan. Meraka hanya dapat mengandalkan komunikasi melalui bahasa isyarat atau tulisan. Bahasa isyarat yang digunakan oleh kaum tunawicara ini sulit dipahami oleh masyarakat pada umumnya, sehingga kaum tunawicara merasa terasingkan bagi lingkungan disekitarnya.

Seiring dengan kemajuan teknologi, telah dilakukan penelitian dalam rangka untuk menghasilkan piranti bantu untuk menerjemahkan bahasa isyarat ke dalam tulisan bahkan suara. Kategori penelitian yang dilakukan dapat dibedakan menjadi dua yaitu pendekatan berbasis visi computer (computer vision) dan pendekatan berbasis sensor. Pada pendekatan berbasis visi computer digunakan file video atau langsung melalui kamera yang menangkap gerakan bahasa isyarat. Pada pendekatan ini sebelum dilakukan proses pengenalan harus dilakukan pra proses berupa pengolahan citra dulu, seperti segmentasi dan tracking tangan. Sedangkan pendekatan dengan sensor terintegrasi dengan sarung tangan (glove) yang menghasilkan besaran listrik yang terukur, untuk mengetahui derajat tekukan jari jari tangan dan gerakan tangan.

Dalam bahasa isyarat, setiap gerakan sudah menugaskan makna, dan aturan yang kuat dari konteks dan tata bahasa dapat diterapkan untuk membuat pengakuan penurut. Bahasa Isyarat Indonesia (BISINDO) adalah bahasa pilihan untuk kebanyakan tuli dan bisu di Indonesia. Sistem Isyarat Bahasa Indonesia (SIBI) menggunakan sekitar 6.000 gerak kata-kata umum dan jari ejaan kata-kata untuk berkomunikasi mengaburkan atau kata benda. Namun, mayoritas penandatanganan adalah dengan kata-kata penuh, memungkinkan percakapan ditandatangani untuk melanjutkan pada tentang laju percakapan lisan. Tata bahasa BISINDO memungkinkan lebih banyak fleksibilitas dalam urutan kata dari bahasa Indonesia dan kadang-kadang menggunakan redundansi untuk penekanan kata atau ejaan.

Untuk memberikan kemudahan terhadap penyandang tunawicara dan tunarungu, maka perlu adanya modul yang bisa membantu penyandang tunawicara untuk berinteraksi dengan orang lain. Dengan menggunakan telepon genggam yang telah dilengkapi dengan kamera dan sistem android untuk menerjemahkan bahasa isyarat. Penyandang tunawicara bisa lebih mudah berinteraksi dengan masyarakat umum. Serta masyarakat umum bisa belajar untuk mengerti arti bahasa yang di isyaratkan penyandang tunarungu dan tunawicara.

1.2. Perumusan Masalah

Permasalahan yang dibahas dalam tugas akhir ini adalah:

1. Proses pendeteksian gesture tangan.
2. Mendapat ekstraksi gesture tangan.
3. Proses tracking pergerakan tangan.
4. Penentuan label dari pergerakan perubahan gerakan tangan.

1.3. Batasan Masalah

Batasan masalah dalam tugas akhir ini adalah

1. Bahasa Isyarat Indonesia yang dikenali adalah yang dibentuk oleh gerakan satu tangan, yaitu tangan kanan.
2. Data uji yang digunakan 26 huruf yang dipilih dari system Bahasa Isyarat Indonesia
3. Lokasi pelaksanaan memiliki pencahayaan yang cukup.
4. Kata yang di uji termasuk kedalam isyarat pokok, yaitu yang melambangkan kata atau konsep.
5. Pengujian hanya dilakukan didalam ruangan dengan kondisi latar belakang homogeny atau relative satu warna.
6. Pengujian tidak dilakukan pada kondisi backlight dan berkaca.

1.4. Tujuan

Tujuan yang ingin dicapai dalam perancangan ini adalah:

1. Mendeteksi dan pengolahan citra deteksi tangan manusia menggunakan kamera
2. Menentukan ekstraksi data hasil pengolahan citra untuk data perubahan gerakan tangan dan akselerasi telapak tangan.
3. Menentukan parameter parameter hand recognition sehingga didapatkan konvergensi dan hasil akurasi yang optimal.

1.5. Metodologi

Langkah-langkah yang dikerjakan pada tugas akhir ini adalah sebagai berikut:

1. **Studi Literatur**

   Pada tahap ini dilakukan pengumpulan dasar teori yang menunjang dalam penulisan Tugas Akhir. Dasar teori ini dapat diambil dari buku-buku, jurnal, dan artikel-artikel di internet dan forum-forum diskusi internet serta survei pada sekolah SLB dan pakar divable

2. **Perancangan *Software***

   Pembuatan *source code* maupun software, diawali dengan cara mengenali ada tidaknya pengguna yang berdiri didepan kamera. Dimana pengenalan dilakukan dengan pengambilan sampling warna kulit pengguna. Sampling ini dilakukan untuk mendapatkan depth pada persendian jari tangan. *Depth* ini digunakan sebagai *scalling* dan referensi jarak objek dengan kamera. Selanjutnya data visual diambil dan dengan metode gesture recognition untuk proses pengenalan anggota tubuh dan proses pendeteksian gerak tangan diambil. Selanjutnya data-data tersebut dicocokkan dan diklarifikasi dengan data pergerakan tangan yang ada. Setelah semua proses tersebut dilakukan makahasil akhirnya akan ditampilkan pada *display mobile phone.*

3. **Perancangan *Hardware***

   Pada tahap ketiga ini akan dilakukan pengkondisian untuk proses uji coba seperti peletakkan kamera sebagai media pengambilan data visual dan koneksinya dengan *processing unit* dan komunikasinya dengan display monitor.

4. **Pengujian Sistem**

   Pengujian alat dilakukan untuk menentukan keandalan dari sistem yang telah dirancang. Pengujian dilakukan untuk melihat apakah *software* dan *hardware* dapat bekerja secara baik.

   Untuk pengujian dapat dilakukan dalam 2 tahap. Pertama adalah tes dengan inisial data gesture yang telah disiapkan diawal dengan kondisi pencahayaan yang ideal yaitu tidak terlalu terang dan tidak terlalu redup sehingga pattern warna dari tangan dapat terdeteksi secara sempurna. Lalu yang kedua yaitu pengujian pada kondisi yang berbeda-beda dimana pengujian

akan dilakukan dengan tiga kondisi pencahayaan yang berbeda serta tiga kondisi latar yang berbeda.

5. **Analisa**

   Data-data yang diperoleh pada tahapan pengujian perlu di analisa lebih lanjut untuk mencari tahu parameter-parameter apa saja yang dapat mempengaruhi proses recognition. Berdasarkan data-data tersebut juga dapat diketahui langkah-langkah pencegahan ataupun mengatasi masalah yang timbul.

6. **Penyusunan Laporan Tugas Akhir**

   Tahap penulisan laporan tugas akhir adalah tahapan terakhir dari proses pengerjaan tugas akhir ini. Laporan tugas akhir berisi seluruh hal yang berkaitan dengan tugas akhir yang telah dikerjakan yaitu meliputi pendahuluan, tinjauan pustaka dan teori penunjang, perancangan sistem, pengujian, dan penutup.

1.6. Sistematika Penulisan

Dalam buku tugas akhir ini, pembahasan mengenai sistem yang dibuat terbagi menjadi lima bab dengan sistematika penulisan sebagai berikut:

- BAB I : PENDAHULUAN

  Bab ini meliputi penjelasan latar belakang, rumusan masalah, batasan masalah, tujuan, metodologi, sistematika penulisan, dan relevansi.

- BAB II : TINJAUAN PUSTAKA DAN TEORI PENUNJANG

  Bab ini menjelaskan tentang teori penunjang dan *literature* yang dibutuhkan dalam pengerjakan tugas akhir ini. Dasar teori yang menunjang meliputi citra digital, *sign language, support vector machine, gesture recognition*, *optical flow, android studio*. Bagian ini memaparkan mengenai beberapa teori penunjang dan beberapa literatur yang berguna bagi pembuatan Tugas Akhir ini.

- BAB III : PERANCANGAN SISTEM

  Bab ini menjelasakan tentang perencanaan sistem baik perangkat keras (*hardware*) maupun perangkat lunak (*software*) untuk sistem penerjemabah bahasa isyarat berupa ASL.

- BAB IV : PENGUJIAN
  Pada bab ini akan menjelaskan hasil uji coba sistem beserta analisanya.

- BAB V : PENUTUP
  Bagian ini merupakan bagian akhir yang berisikan kesimpulan yang diperoleh dari pembuatan Tugas Akhir ini, serta saran-saran untuk pengembangan lebih lanjut.

1.7. Relevansi dan Manfaat

Hasil yang diharapkan dari tugas akhir ini diharapkan mampu memudahkan masyarakat untuk belajar memahami bahasa isyarat berupa perubahan gerak tanga yang dilakukan oleh orang tunawicara. Pengembangan lebih lanjut dari sistem ini adalah penambahan filter background sehingga pendektesian gesture tangan lebih akurat.

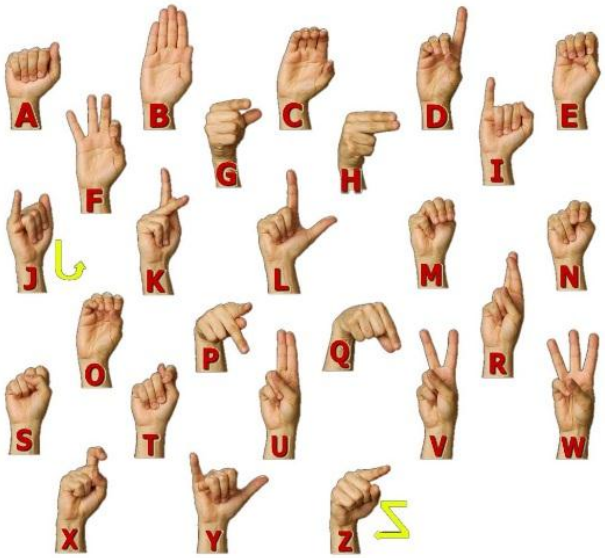
Halaman ini sengaja dikosongkan

BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

Tinjauan pustaka dalam bab ini menjelaskan mengenai system-sistem yang berhubungan tugas akhir ini serta telah diimplementasikan oleh penulis-penulis lain. Sedangkan dasar teori menjelaskan tentang teori yang mendukung keseluruhan sistem pada tugas akhir ini.

2.1 Bahasa Isyarat

Bahasa isyarat merupakan bahasa yang mengutamakan komunikasi manual, yaitu menggunakan bahasa tubuh, tangan dan gerak bibir, bukan suara lisan. Kaum tunarungu adalah kelompok utama yang menggunakan bahasa ini, biasanya dengan mengkombinasikan bentuk tangan, orientasi dan gerak tangan, lengan, dan tubuh, serta ekspresi wajah untuk mengungkapkan pikiran mereka.



Gambar 2.1 Ilustrasi bentuk gerakan tangan untuk mengimplentasikan sebuah huruf [1]

Bahasa isyarat unik dalam jenisnya di setiap negara. Bahasa isyarat bisa saja berbeda di negara-negara yang berbahasa sama. Contohnya, Amerika Serikat dan Inggris meskipun memiliki bahasa tertulis yang sama, memiliki bahasa isyarat yang sama sekali berbeda (American Sign Language dan British Sign Language)[1].

Untuk di negara Indonesia sendiri, sistem yang sekarang umum digunakan adalah Sistem Isyarat Bahasa Indonesia (SIBI) dimana sistem ini sama dengan bahasa isyarat yang diterapkan di Amerika (ASL - American Sign Language). SIBI diciptakan dengan beberapa alasan, di antaranya untuk merepresentasikan Bahasa Indonesia pada tangan, untuk mengajarkan Bahasa Indonesia secara yang sesuai dengan Ejaan Yang Disempurnakan (EYD), dan karena mudah dipelajari oleh orang yang sudah bisa berbahasa Indonesia

2.2 Pengolahan Citra Digital

Pengolahan Citra merupakan proses pengolahan dan analisis citra yang banyak melibatkan persepsi visual. Proses ini mempunyai ciri data masukan dan informasi keluaran yang berbentuk citra. Istilah pengolahan citra digital secara umum didefinisikan sebagai pemrosesan citra dua dimensi dengan komputer. Dalam definisi yang lebih luas, pengolahan citra digital juga mencakup semua data dua dimensi. Citra digital adalah barisan bilangan nyata maupun kompleks yang diwakili oleh bit-bit tertentu.

Pengolahan citra bertujuan untuk memperbaiki kualitas gambar, dilihat dari aspek radiometric dan aspek geometric. Aspek radiometric terdiri dari peningkatan kontras, restorasi citra, transformasi warna sedangkan aspek geometric terdiri dari rotasi, skala, translasi, trnsformasi geometric). Yang kedua untuk melakukan proses penarikan informasi atau deskripsi obyek atau pengenalan obyek yang terkandung pada citra. Lalu, untuk melakukan pemilihan citra ciri (feature images) yang optimal untuk tujuan analisis. Dan yang terakhir untuk melakukan kompresi atau reduksi data untuk tujuan penyimpanan data, transmisi data, dan waktu proses data.

2.2.1 Model Citra Digital

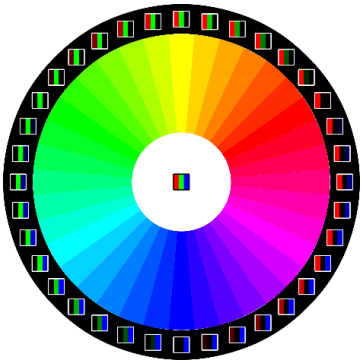
Untuk menampilkan sebuah gambar pada layar-layar digital, gambar tersebut harus melalui proses digitalisasi citra untuk diubah menjadi citra digital. Proses-proses tersebut meliputi proses sampling dan proses *quantization*. Proses sampling membagi gambar menjadi beberapa kotak kecil. Proses *quantization* memberi nilai kecerahan dari setiap kotak tersebut. Nilai hasil *quantization* juga sering disebut nilai kedalaman.

Hasil dari proses digitalisasi citra adalah sebuah matriks yang berisi nilai-nilai riil. Elemen-elemen dari matriks tersebut disebut *picture element* atau sering disebut pixel. Sebuah pixel mempunyai dua property yaitu koordinat posisi dari pixel tersebut dan nilai dari pixel tersebut[1]. Sehingga, sebuah pixel dapat dinyatakan sebagai fungsi dua dimensi $f(x, y)$ dengan titik asal x dan y ada di pojok kiri atas dari citra. Misalkan, sebuah pixel dinyatakan $f(2, 3) = 5$, berarti pixel tersebut berada di posisi $x = 2$ dan $y = 3$ dari pojok kiri atas citra dengan tingkat kecerahan 5. Gambar 2.1 menunjukkan citra yang dilihat oleh manusia merupakan kumpulan matriks data yang dibaca oleh komputer. Dari gambar tersebut bisa dilihat bahwa, pada potongan citra bagian spion mobil memiliki nilai kecerahan sebesar 194 pada koordinat $x = 0$ dan $y = 0$. Sedangkan pada koordinat $x = 1$ dan $y = 0$, kecerahan citra adalah 210.

Kedalaman pixel sering disebut juga kedalaman warna. Citra digital yang memiliki kedalaman pixel n bit disebut juga citra n-bit. Dalam pemrosesan citra digital, citra digital bisa dibedakan berdasarkan batas nilai kecerahan atau nilai kedalaman dari suatu citra. Beberapa jenis citra berdasarkan kedalaman citra, di antaranya adalah citra warna atau sering disebut sebagai citra RGB (Red Green Blue), citra grayscale, dan citra biner.

2.2.1.1 Model Citra RGB

Citra warna adalah citra yang setiap pixel-nya ditentukan oleh tiga nilai masing-masing untuk komponen merah, biru, dan hijau[2]. Komponen warna lebih dikenal sebagai kanal. Sehingga, nilai dalam satu pixel-nya terdapat 3 kanal yang mewakili nilai dari warna merah, warna hijau, dan warna biru.



Gambar 2.2 Pattern Warna

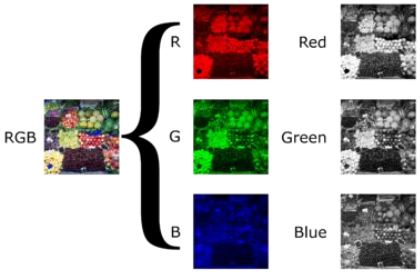
Kombinasi nilai dari ketiga warna tersebut akan membentuk suatu warna tertentu. Gambar 2.2 menunjukkan warna-warna yang dihasilkan dari kombinasi nilai warna merah, hijau, dan biru.

Nilai dari masing-masing komponen merah, hijau, dan biru memiliki rentang tertentu, tergantung tipe data yang dipakai. Jika tipe data tiap kanal warna tersebut berupa tipe data 8 bit, rentang nilai setiap kanal berada pada rentang nilai antara 0 sampai 255. Jika tipe datanya 16 bit, rentang nilai setiap kanal adalah antara 0 sampai 65535. Jumlah bit yang lebih banyak akan memberi variasi warna yang lebih banyak pula. Tipe data 8 bit per kanal akan memberi variasi warna sebanyak 16777216 warna, sedangkan tipe data 16 bit per kanal akan membentuk 281474976710656 jenis warna. Akan tetapi, dalam proses pengolahan citra, jumlah bit yang lebih sedikit memberi keuntungan pengolahan citra yang lebih cepat.[3]

2.2.1.2 Model Citra Greyscale

Mengubah citra RGB menjadi citra grayscale digunakan untuk menyederhanakan model citra, sehingga waktu yang dibutuhkan untuk melakukan proses selanjutnya berjalan lebih cepat. Untuk mendapatkan citra grayscale adalah :

$$S = \frac{r+g+b}{3} \tag{2.1}$$



Gambar 2.3 Konversi RGB layer pada citra berwarna ke Grayscale[3]

2.2.1.3 Model Citra Biner

Citra biner merupakan array logika yang hanya berisi 0 dan 1, yang diartikan hitam dan putih. Untuk citra biner yang normal, objek biasanya ditandai dengan warna hitam. Sedangkan warna putih menunjukkan background.

Citra biner berasal dari citra grayscale yang di-threshold. Sebagai contoh, suatu citra grayscale di-threshold menggunakan nilai threshold 100, maka pixel yang memiliki nilai gray level kurang dari 100 akan diberi nilai 0 (warna hitam), sedangkan pixel yang memiliki nilai gray level di atas 100 akan diberi nilai 255 (warna putih). Gambar 2.4 menunjukkan hasil konversi menjadi citra biner.

Gambar 2.4 Citra Biner [3]

Walaupun citra warna lebih disukai banyak orang, namun citra biner tetap dipakai, terutama dalam pemrosesan citra. Hal ini dikarenakan kebutuhan memori citra biner kecil sehingga waktu pemrosesan lebih cepat dibandingkan dengan citra grayscale ataupun warna.

2.2.2 Segmentasi Citra Berdasarkan Warna

Segmentasi citra akan membagi-bagi suatu citra menjadi daerah-daerah atau obyekobyek yang dimilikinya. Menurut Castleman (1996) segmentasi citra merupakan suatu proses memecah suatu citra digital menjadi banyak segmen/bagian daerah yang tidak saling bertabrakan (nonoverlapping). Dalam konteks citra digital daerah hasil segmentasi tersebut merupakan kelompok piksel yang bertetangga atau berhubungan. Segmentasi citra dapat dilakukan melalui beberapa pendekatan, menurut Castleman (1996) terdapat 3 macam pendekatan, antara lain:

- Pendekatan batas (boundary approach), pendekatan ini dilakukan untuk mendapatkan batas yang ada antar daerah.
- Pendekatan tepi (edge approach), pendekatan tepi dilakukan untuk mengidentifikasi piksel tepi dan menghubungkan piksel-piksel tersebut menjadi suatu batas yang diinginkan
- Pendekatan daerah (region approach), pendekatan daerah bertujuan untuk membagi citra dalam daerah-daerah sehingga didapatkan suatu daerah sesuai kriteria yang diinginkan.

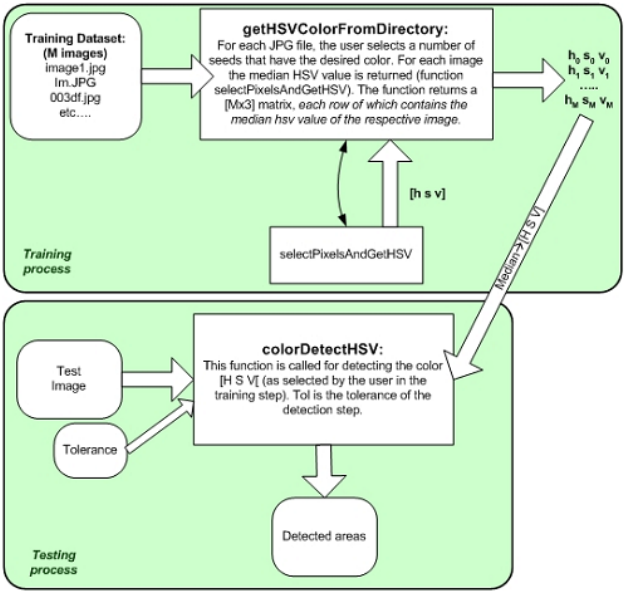
Proses segmentasi digunakan dalam berbagai penerapan, meskipun metode yang digunakan sangat bervariasi, semuanya memiliki tujuan sama: mendapatkan representasi sederhana yang berguna dari suatu citra. Terdapat berbagai macam metode dalam melakukan segmentasi, cukup sulit untuk menentukan metode yang komprehensif, oleh karena itu pemilihan metode bergantung pada pendekatan yang akan digunakan dan fitur yang ingin diperoleh dari citra

2.2.2.1 Segmentasi pada Ruang Warna HSV

Segmentasi warna merupakan proses segmentasi dengan pendekatan daerah yang bekerja dengan menganalisis nilai warna dari tiap piksel pada citra dan membagi citra tersebut sesuai dengan fitur yang diinginkan. Segmentasi citra dengan deteksi warna HSVmenurut Gunanto

(2009) menggunakan dasar seleksi warna pada model warna HSVdengan nilai toleransi tertentu.

Pada metode segmentasi dengan deteksi warna HSV menurut Giannakupoulos (2008), dilakukan pemilihan sampel piksel sebagai acuan warna untuk membentuk segmen yang diinginkan. Citra digital menggunakan model warna RGB sebagai standar acuan warna, oleh karena itu proses awal pada metode ini memerlukan konversi model warna RGBke HSV. Untuk membentuk segmen sesuai dengan warna yang diinginkan maka ditentukan nilai toleransi pada setiap dimensi warna HSV, kemudian nilai toleransi tersebut digunakan dalam perhitungan proses adaptive threshold. Hasil dari proses thresholdtersebut akan membentuk segmen area dengan warna sesuai toleransi yang diinginkan. Secara garis besar, gambaran proses segmetnasi dapat dilihat pada Gambar 1 dan berikut ini merupakan proses segmentasi menurut Giannakupoulos (2008).



Gambar 2.5 Skema Deteksi Warna HSV

- Tentukan citra RGByang menjadi obyek deteksi, nilai warna HSVyang menjadi acuan (hasil proses pelatihan data) dan nilai toleransi HSVyang digunakan.
- Transpose citra RGBke HSV
- Lakukan filter warna pada citra berdasarkan nilai acuan (T) dan nilai toleransi (tol). Dengan x sebagai warna HSVpada piksel yang ada maka warna yang tidak termasuk dalam rentang T-tol < x < T+toldiberi warna hitam.
- Transpose kembali citra ke RGB, tampilkan hasil filter.

Gunanto (2009) mencoba melakukan segmentasi dengan menggunakan metode deteksi warna HSVpada bagian tubuh manusia yang diberi warna tertentu. Gunanto melakukan penelitian berdasarkan metode deteksi warna HSVGiannakupoulos untuk memisahkan bagian tubuh manusia yang diberi warna tertentu. Penelitian tersebut menggunakan data input berupa citra 2 dimensi dengan latar belakang homogen berwarna putih dan fitur model manusia. Model manusia mengenakan pakaian dengan warna tertentu pada tiap bagian tubuh manusia yang akan dikenali. Proses segmentasi warna dengan menggunakan deteksi warna HSV menghasilkan segmen warna yang akurat sesuaidengan warna sampel dan nilai toleransi yang diberikan. Hasil segmentasi warna tersebut menghasilkan segmen citra yang membentuk suatu blob,yaitu sekumpulan piksel bertetangga yang memiliki nilai tertentu.

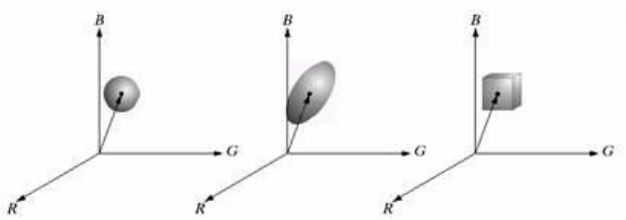
2.2.2.2 Segmentasi pada Ruang Vektor RGB

Meski ada HSI namun segmentasi mendapatkan hasil lebih baik dengan menggunakan RGB Color Vector. Misal kita akan menyegmen obyek dengan range warna tertentu pada RGB image. Diberikan contoh himpunan warna yang dinginkan, maka kita dapatkan rata-rata estimasi warna yang diinginkan. Warna rata-rata ini menunjukkan RGB vector a. Tujuan segmentasi untuk menspesifkasi pixel RGB pada image apakah memiliki warna yang ada dalam range warna atau tidak .dalam melakukan perbandingan, pentinhg untuk mengukur kesamaan. Pengukuran yang paling mudah menggunakan jarak euclidean. z menunjukkan titik

arbitrary pada RGB space. Katakan z mirip dengan a, bila jaraknya kurang dari specified threshold, D0. Jarak euclidean antara z dan a adalah

$$\begin{aligned} D(\mathbf{z}, \mathbf{a}) &= \|\mathbf{z} - \mathbf{a}\| \\ &= [(\mathbf{z} - \mathbf{a})^T(\mathbf{z} - \mathbf{a})]^{\frac{1}{2}} \\ &= [(z_R - a_R)^2 + (z_G - a_G)^2 + (z_B - a_B)^2]^{\frac{1}{2}} \end{aligned} \tag{2.2}$$

R,G,B menunjukkan komponen RGB pada vektor a dan z. Letak titik titik dimana D(z,a) <= D0 merupakan bola pejal dengan radius D0.Fig 6.43.



Gambar 2.6 Daerah data untuk segmentasi vector RGB

2.2.2.3 Deteksi Tepi Warna

Deteksi tepi pada suatu citra adalah suatu proses yang menghasilkan tepi-tepi objek citra. Suatu tepi muncul ketika ada dua daerah yang mempunyai perbedaan intensitas pada suatu citra. Dalam satu dimensi, tepi dihubungkan dengan puncak lokal pada pada fungsi turunan pertama dari intensitas.

Gradien adalah hasil pengukuran perubahan dalam sebuah fungsi intensitas. Gradien didefinisikan sebagai vektor karena menujukan arah penambahan laju maksimum fungsi f(x,y) dan besaran gradien yang sesuai dengan fungsi f(x,y)

2.2.3 Kompresi Citra

Sebutan kompresi data mengacu pada pengurangan data yang dibutuhkan untuk menampilkan sebuah informasi. Sebuah perbedaan yang jelas harus dibuat antara data dan informasi. Beragam data dapat digunakan untuk menampilkan sejumlah informasi yang sama. Contohnya, bila ada dua macam individu, yang pertama adalah subyek

yang sangat berbelit-belit dan subyek kedua adalah orang yang sangat singkat dan to-the-point, kedua subyek ini diminta untuk menceritakan sebuah cerita yang sama. Bila kedua subyek ini menggunakan jumlah kata yang berbeda untuk menceritakan dasar cerita yang sama, dua versi cerita yang berbeda akan tercipta, dan sedikitnya mengikutkan sebuah data yang tidak penting, yaitu berisi data (atau kata-kata) yang berisi informasi yang tidak relevan atau mengulang informasi yang telah diketahui sebelumnya. Inilah disebut data redundancy.

Data redundancy adalah sebuah pembicaraan sentral pada kompresi citra digital. Hal ini bukanlah konsep yang abstrak tapi sebuah entitas yang dapat dihitung secara matematis. Bila n1 dan n2 berarti jumlah unit pembawa informasi dalam dua kumpulan data yang merepresentasikan informasi yang sama, maka relative data redundancy RD dari kumpulan data yang pertama (n1) dapat didefinisikan sebagai

$$R_D = 1 - \frac{1}{C_R} \tag{2.3}$$

Dimana CR, yang umumnya dipanggil rasio kompresi, adalah

$$C_R = \frac{n_1}{n_2} \tag{2.4}$$

Dalam kompresi citra digital, tiga dasar perulangan data (data redundancy) dapat diidentifikasi dan dieksploitasi menjadi: coding redundancy, interpixel redundancy, dan psychovisual redundancy. Kompresi data dijangkau saat satu atau lebih dari perulangan-perulangan ini dikurangi atau dihilangkan.

2.2.3.1 Model Kompresi Citra

Kita telah diskusikan diatas, tiga teknik umum untuk mengurangi atau mengkompresi jumlah dari data yang dibutuhkan menampilkan sebuah citra. Bagaimanapun, teknik tersebut secara tipikal dikombinasikan untuk membentuk sistem kompresi yang praktis. Dalam sub-bab ini, kita menguji semua karakteristik dari sistem itu dan mengembangkan sebuah model untuk menampilkannya.

Enkode dan decode digunakan disini. Sebuah citra input f(x,y) dimasukkan kedalam encoder, yang menciptakan sebuah kumpulan symbol dari data input. Setelah transimisi lintas jalur,

representasi yang telah di-encode dimasukkan kedalam decoder, dimana sebuah output citra yang telah direkonstruksi dihasilkan. Secara umum, f(x,y) mungkin atau tidak mungkin menjadi replica dari f(x,y).

Baik encoder maupun decoder terdiri atas dua fungsi relatif yang independen atau sub-blok. Encoder terbuat dari sebuah sumber encoder, yang menghilangkan perulangan input, dan sebuah jalur encoder / channel encoder, yang menambah kekebalan noise / gangguan dari sumber output encoder. Sedangkan decoder terdiri atas sebuah jalur decoder / channel decoder diikuti oleh sebuah sumber decoder. Bila jalur antara encoder dan decoder bebas gangguan (tidak rentan error), jalur encoder dan decoder diabaikan, dan encoder dan decoder umum menjadi sumber encoder dan decoder, secara umum.

- **Encoder dan Decoder Sumber**

  Sumber encoder bertanggungjawab untuk mengurangi atau menghilangkan segala perulangan kode, interpixel, atau psychovisual pada citra input. Aplikasi spesifik dan kebutuhan kejelasan yang berasosiasi mendikte pendekatan encoding terbaik untuk digunakan pada segala situasi. Secara normal, pendekatan tersebut dapat dimodelkan dengan sebuah seri dari tiga operasi independen. Pada tahap pertama dari proses sumber encoding, mapper / pemeta mentransformasikan data input kedalam sebuah format yang didesain untuk mengurangi pengulangan interpixel pada citra input. Operasi ini umumnya dua arah dan mungkin ataupun tidak mungkin secara langsung mengurangi jumlah dari data yang dibutuhkan untuk menampilkan citra.

- **Jalur Encoder dan Decoder**

  Jalur encoder dan decoder memainkan peran yang penting dalam keseluruhan proses encoding-decoding saat jalur pada gambar 8.5 terganggu atau rentan error. Mereka didesain untuk mengurangi dampak dari gangguan jalur dengan menyisipkan sebuah bentuk terkontrol dari perulangan kedalam sumber data ter-encode.

  Salah satu teknik jalur encoding yang paling berguna ditemukan oleh R.W. Hamming (1950). Teknik ini berdasarkan pada penambahan bit secukupnya kepada data yang sedang di-

encode untuk menjamin bahwa beberapa angka minimum dari bit-bit harus berubah diantara kata-kata kode yang valid. Hamming menunjukkan, sebagai contoh, bahwa bila 3 bit dari perulangan ditambahkan kepada kata 4-bit, sehingga jarak antara dua kode kata valid manapun adalah 3, semua kesalahan bit-tunggal dapat dideteksi dan dikoreksi.

2.2.4 Pengolahan Morfologi Citra

Nama morfologi secara matematis didapatkan dari pembelajaran tentang bentuk. Pendekatan ini menggali fakta bahwa di dalam banyak aplikasi bahasa mesin adalah wajar, dan mudah untuk berfikir dalam bentuk saat merancang algoritma. Pendekatan morfologikal memfasilitasi pemikiran berbasiskan bentuk atau ikonis. Dalam pendekatan morfologi, unit yang paling dasar dari informasi bergambar adalah gambar biner.

Morfologi matematis mendeskripsikan suatu aplikasi operator aljabar, yang dapat digunakan untuk mengambil suatu daerah, bentuk dan batas pada gambar.

2.2.4.1 Erosi dan Dilatasi

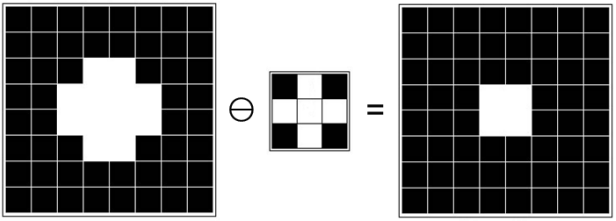
- **Erosi**

Erosi merupakan proses untuk mengecilkan ataupun menipiskan objek pada gambar. Sama seperti dilatasi, proses ini juga dipengaruhi oleh structuring element yang digunakan pada pengolahan gambar.

Definisi matematis erosi mirip dengan dilatasi. Dimana erosi A oleh B, dinotasikan $A \ominus B$, didefinisikan sebagai persamaan .

$$A \ominus B = \{Z \mid \llbracket (B) \rrbracket \_X \cap A \hat{} C \neq \varphi\} \qquad (2.5)$$

Secara algoritma kita dapat mendefinisikan erosi sebagai : Pertama-tama tiap nilai yang ada pada keluaran $A \ominus B$ akan dijadikan nilai 0 (zero). B kemudian akan ditempatkan pada tiap lokasi A yang memiliki nilai pixel 0 (dengan mengacu pada titik referensi B). Bila pada A terdapat B (pada proses A AND B tidak sama dengan zero) maka pada keluaran akan digunakan nilai B. Hasil keluaran akhir adalah setiap elemen dari A yang tidak terkena elemen B pada proses diatas. Untuk lebih jelasnya dapat dilihat pada gambar dibawah.



Gambar 2.7 Ilustrasi Morfologi Erosi (a) merupakan gambar awal, dengan objek bernilai 1; (b) structuring element; (c) hasil akhir proses erosi.

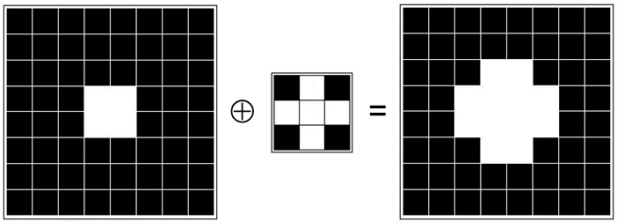
- **Dilatasi**

Dilatasi merupakan proses menambahkan atau mempertebal objek pada gambar biner. Hal ini dilakukan dengan memberikan tambahan pixel pada batasan suatu objek pada suatu gambar. Jumlah pixel yang ditambahkan tergantung dari ukuran dan bentuk dari stucturing element yang digunakan pada pengolahan gambar. Dimana stucturing element, merupakan sebuah matrik yang hanya berisi 1 dan 0, yang dapat memiliki berbagai bentuk dan ukuran.Secara matematis dilatasi A oleh B, dinotasikan A⊕B, didefinisikan sebagai persamaan.

$$A \oplus B = \{Z \mid [\![(B)]\!]\_X \cap A \hat{} C \neq \varphi\} \qquad (2.6)$$

Dimana φ merupakan set yang kosong dan B adalah stucturing element-nya. Atau dapat juga dikatakan sebagai dilatasi A oleh B merupakan suatu set yang didalamnya terdiri dari semua stucturing element dari lokasi awalnya, dimana B yang telah direfleksikan, dan ditranslasikan akan tumpang tindih setidaknya dengan sebagian dari gambar A.

Secara algoritma operasi ini dapat didefinisikan sebagai : Anggaplah bahwa B merupakan sebuah mask. Kemudian taruhlah titik referensi dari stucturing element pada pixel yang memiliki nilai 1 pada gambar A. dan kemudian lakukan fungsi OR pada A dan B. Sehingga bila terdapat nilai yang berbeda pada pixel A dan B yang memiliki lokasi yang sama, maka nilai dari keluarannya adalah 1. Namun bila bilai pada pixel A dan B sama maka nilai keluarannya akan sama dengan nilai terebut. Proses ini kemudian akan diulang terus menerus, hingga nilai titik referensi B telah berada di semua titik A yang memiliki nilai 1. Untuk lebih jelasnya dapat dilihat pada gambar 2.8 .



Gambar 2.8 Ilustrasi Morfologi Dilatasi(a) merupakan gambar awal, dengan objek bernilai 1; (b) structuring element; (c) hasil akhir proses dilatasi.

Dilatasi pada umumnya memperbesar ukuran suatu objek, menutupi lubang, dan menghubungkan area-area yang lebih kecil dari struktur elemennya.

2.2.4.2 Opening dan Closing Citra

- **Morfologi Opening Citra**

Opening merupakan suatu proses dimana suatu gambar yang terlebih dahulu mengalami proses erosi, disusul dengan proses dilatasi dengan menggunakan struktur elemen yang sama. Hal ini bertujuan untuk memulihkan sebesar mungkin data yang telah hilang karena erosi pada suatu gambar. Selain itu erosi juga dapat digunakan untuk menghilangkan objek yang terlalu kecil, memisahkan antar objek yang jaraknya berdekatan.

Opening A oleh B, yang didenotasikan dengan $A \circ B$, didefinisikan sebagai persamaan 2.18.

$$A \circ B = (A \ominus B) \oplus B \tag{2.7}$$

Sehingga $A \circ B$, merupakan gabungan dari semua translasi B yang termasuk dalam A. Opening membuang semua bagian dari objek yang tidak dapat menampung structuring element-nya. Sehingga memperhalus bagian tepi objek tanpa merusak bentuk aslinya, dan mengilangkan objek yang terlalu kecil atau tipis.

Gambar 2.9 Ilustrasi Morfologi Opening[3] (a) gambar asli; (b) gambar asli yang telah mengalami erosi; (c) gambar (b) yang telah didilatasi dengan structuring element yang sama.

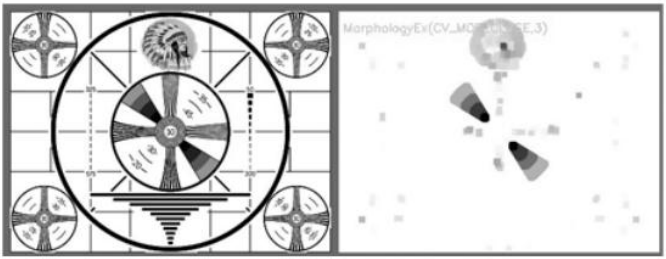
- **Morfologi Closing Citra**

Closing merupakan kebalikan dari *opening*, dimana suatu gambar mengalami proses dilatasi terlebih dahulu sebelum kemudian dierosi dengan struktur elemen yang sama. Operasi ini biasanya digunakan untuk menghilangkan *noise* yang ada pada gambar.

Operasi *closing* A oleh B, yang didenotasi sebagai A·B, didefinisikan sebagai persamaan 2.18.

$$A \cdot B = (A \oplus B) \ominus B \tag{2.8}$$

Closing A·B merupakan komplemen dari semua gabungan dari semua translasi *B* yang tidak tumpang tindih dengan *A*. Hal ini memungkinkan operasi ini untuk menutup lubang kecil atau garis tipis pada suatu objek, dan menghubungkan 2 objek yang letaknya berdekatan, serta menghaluskan batas tiap objek tanpa merusak bentuk aslinya.



Gambar 2.10 Ilustrasi Morfologi Closing[3] dimana garis-garis hitam (latar) menjadi lebih sedikit, sedangkan bagian putih (objek) membesar.

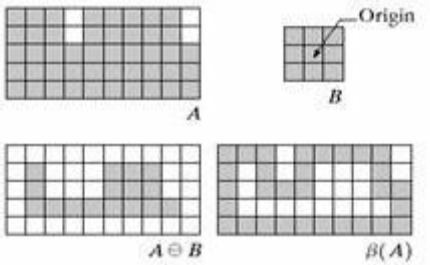
2.2.4.3 Algoritma Dasar Morfologi Citra

- **Boundary Extraction**

Boundary dari himpunan A bisa didapatkan dengan:

$$\beta(A) = A - (A \ominus B) \tag{2.9}$$

B adalah "structuring element" yang sesuai.



Gambar 2.11 Ilustrasi Proses Boundary (a). Set A, (b). Stuktur Elemen dari B, (c). A dierosi oleh B, (d).Boundary dari Set A dan Erosi A oleh B

Gambar 2.12 Ilustrasi penggunaan Boundary untuk membuat silhouette tangan manusia[6]

Ekstraksi Boundary juga bisa digunakan untuk membuat silhouette bentuk tangan mabusi. Dimanada model ini tidak menggunakan representasi spasial tubuh lagi, karena mereka memperoleh parameter langsung dari gambar atau video menggunakan template database. Beberapa didasarkan pada template 2D *deformable* bagian tubuh, terutama tangan manusia. Template deformable yang set poin pada garis

dari objek, digunakan sebagai sisipan node untuk objek garis besar perkiraan. Salah satu fungsi interpolasi yang paling sederhana linear, yang melakukan bentuk rata-rata dari titik set, parameter variabilitas titik dan perusak pendidikan eksternal. Model berbasis template ini banyak digunakan untuk tangan-pelacakan, tetapi juga dapat digunakan untuk klasifikasi sikap sederhana.

Pendekatan kedua dalam gerakan mendeteksi menggunakan model berbasis penampilan menggunakan urutan gambar sebagai gerakan template. Parameter untuk metode ini adalah baik gambar sendiri, atau fitur tertentu berasal dari ini. Sebagian besar waktu, hanya satu (*monoscopic*) atau dua (*stereoskopis*) pandangan yang digunakan.

- **Region Filling**

Dimulai dari satu titik p di dalam boundary, tujuannya adalah mengisi keseluruhan region dengan 1.Jika digunakan konvensi bahwa semua titik non-boundary dilabeli 0, maka kita memberi nilai 1 pada p untuk memulai. Prosedur berikut akan mengisi region dengan nilai 1 :

$$X_k = (X_{k-1} \oplus B) \cap A^c \qquad k = 1,2,3,... \tag{2.10}$$

dengan X0=p, dan B adalah "symmetric structuring element". Algoritma berhenti pada iterasi ke k, jika Xk=Xk-1. Union Xk dan A adalah himpunan isi region dan boundary-nya.

- **Ekstraksi "Connected Component"**

Ekstraksi "connected component" dalam citra biner adalah sentral dari beberapa aplikasi analisis citra terotomasi.

Misal Y menyatakan "connected component" yang berada dalam himpunan A dan diasumsikan bahwa titik p dari Y diketahui. Ekspresi iteratif berikut akan menghasilkan semua elemen Y:

$$X_k = (X_{k-1} \oplus B) \cap A \qquad k = 1,2,3,... \tag{2.11}$$

- **Convex Hull**

Himpunan A dikatakan convex jika garis lurus yang menghubungkan sembarang dua titik di dalam A terletak seluruhnya di A. Convex hull H dari sembarang himpunan S adalah himpunan convex terkecil yang berisi S. Himpunan selisih H-S disebut “convex deficiency” dari S.

Cara ini juga berguna untuk memahami bentuk objek atau kontur adalah untuk menghitung hull convex untuk objek dan kemudian menghitung sifat convexity defect. Bentuk kompleks dari banyak objek juga ditandai dengan defect tersebut. Gambar dibawah menggambarkan konsep convexity defects menggunakan gambar tangan manusia. convexity defects digambarkan sebagai garis gelap di sekitar tangan, dan daerah-daerah yang berlabel A sampai H adalah setiap "defect" relatif terhadap hull itu. Seperti yang anda lihat, convexity defects ini menawarkan sarana mencirikan tidak hanya tangan sendiri, tetapi juga pusat tangan.

Gambar 2.13 Ilustrasi Convex Hull untuk menghubungkan ujung-ujuang jari [5]

Ada tiga metode penting OpenCV yang berhubungan dengan complex hulls dan convexity defects. Pertama hanya menghitung hulls of a contour yang kami telah mengidentifikasi, dan kedua memungkinkan kita untuk memeriksa apakah kontur diidentifikasi sudah convex. Ketiga menghitung sifat convexity defects dalam kontur yang convex hull yang dikenali.

- **Thinning**

Thinning dari himpunan A dengan "structuring element" B didefinisikan dengan :

$$A \otimes B = A - (A * B) = A \cap (A * B)^c \tag{2.12}$$

Seperti pada kasus penentuan convex hull, kita hanya tertarik pada pencocokan pola dengan "structuring elements", jadi tidak diperlukan operasi background dengan transformasi "hit-or-miss". Ekspresi untuk melakukan thinning secara simetris terhadap A dilakukan dengan serangkaian "structuring element" berikut :

$$\{B\} = \{B^1, B^2, B^3, \ldots, B^n\} \tag{2.13}$$

dengan Bi adalah versi Bi-1 yang sudah dirotasi. Kita bisa mendefinisikan thinning menggunakan serangkaian "structuring elements" berikut :

$$A \otimes \{B\} = ((\ldots((A \otimes B^1) \otimes B^2)\ldots) \otimes B^n) \tag{2.14}$$

Prosesnya adalah menguruskan A satu pass dengan B1, selanjutnya menguruskan hasilnya satu pass dengan B2, dan selanjutnya, sampai A dikuruskan satu pass dengan Bn. Keseluruhan proses diulang sampai tidak ada lagi perubahan yang terjadi.

2.3 Gesture Recognition

Gesture recognition adalah topik ilmu pengetahuan dan bahasa dengan tujuan untuk menafsirkan gerakan manusia melalui algoritma matematika. Gerakan dapat berasal dari gerak tubuh tetapi umumnya berasal dari wajah atau tangan. Banyak pendekatan yang telah dibuat menggunakan kamera dan komputer visi algoritma untuk menafsirkan

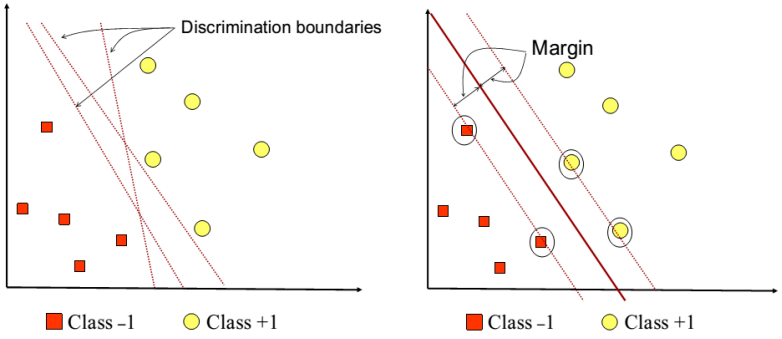
bahasa isyarat. Namun, identifikasi dan pengakuan dari postur, kiprah, proxemics, dan perilaku manusia yang juga merupakan subjek sikap pengakuan teknik. Gesture recognition dapat dilihat sebagai cara untuk komputer untuk mulai memahami bahasa tubuh manusia, dengan demikian membangun jembatan antara mesin dan manusia lebih kaya daripada teks primitif user interface atau bahkan GUI (graphical user interface), yang masih membatasi mayoritas input ke keyboard dan mouse[4].

Gesture recognition memungkinkan manusia untuk berkomunikasi dengan mesin (HMI) dan berinteraksi secara alami tanpa perangkat mekanik. Menggunakan konsep gerakan pengakuan, dimungkinkan untuk menunjuk jari di layar komputer sehingga kursor akan bergerak sesuai. Ini bisa berpotensi membuat perangkat input konvensional seperti mouse, keyboard dan layar sentuh bahkan berlebihan. Gerakan pengakuan dapat dilakukan dengan teknik dari computer vision dan pengolahan citra.

2.4 Support Vector Machine

Support Vector Machine (SVM) pertama kali diperkenalkan oleh Vapnik pada tahun 1992 sebagai rangkaian harmonis konsep-konsep unggulan dalam bidang pattern recognition. Sebagai salah satu metode pattern recognition, usia SVM terbilang masih relatif muda. Walaupun demikian, evaluasi kemampuannya dalam berbagai aplikasinya menempatkannya sebagai state of the art dalam pattern recognition, dan dewasa ini merupakan salah satu tema yang berkembang dengan pesat[7], [8]. SVM adalah metode learning machine yang bekerja atas prinsip Structural Risk Minimization (SRM) dengan tujuan menemukan hyperplane terbaik yang memisahkan dua buah class pada input space. Tulisan ini membahas teori dasar SVM dan aplikasinya dalam bioinformatika, khususnya pada analisa ekspresi gen yang diperoleh dari analisa microarray.

2.4.1 Pattern Recognition



Gambar 2.14 SVM berusaha menemukan hyperplane terbaik yang memisahkan kedua class –1 dan +1 [9]

Pattern Recognition merupakan salah satu bidang dalam komputer sains, yang memetakan suatu data ke dalam konsep tertentu yang telah didefinisikan sebelumnya. Konsep tertentu ini disebut classatau category. Aplikasi pattern recognition sangat luas, di antaranya mengenali suara dalam sistem sekuriti, membaca huruf dalam OCR, mengklasifikasikan penyakit secara otomatis berdasarkan hasil diagnosa kondisi medis pasien dan sebagainya. Berbagai metode dikenal dalam pattern recognition, seperti linear. discrimination analysis, hidden markov model hingga metode kecerdasan buatan seperti artificial neural network. Salah satu metode yang akhir-akhir ini banyak mendapat perhatian sebagai state of the artdalam pattern recognition adalah Support Vector Machine (SVM) . Support Vector Machine (SVM) dikembangkan oleh Boser, Guyon, Vapnik, dan pertama kali dipresentasikan pada tahun 1992 di Annual Workshop on Computational Learning Theory. Konsep dasar SVM sebenarnya merupakan kombinasi harmonis dari teori-teori komputasi yang telah ada puluhan tahun sebelumnya, seperti margin hyperplane (Duda & Hart tahun 1973, Cover tahun 1965, Vapnik 1964, dsb.), kernel diperkenalkan oleh Aronszajn tahun 1950, dan demikian juga dengan konsep-konsep pendukung yang lain. Akan tetapi hingga tahun 1992,

belum pernah ada upaya merangkaikan komponen-komponen tersebut[10].

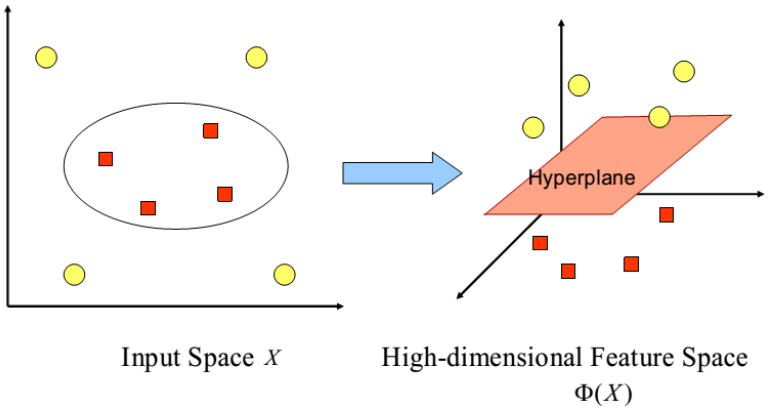
2.4.2 Kernel Trick dan Non Linear Classification SVM

Pada umumnya masalah dalam domain dunia nyata (real world problem) jarang yang bersifat linear separable. Kebanyakan bersifat non linear. Untuk menyelesaikan problem non linear, SVM dimodifikasi dengan memasukkan fungsi Kernel.

Dalam non linear SVM, pertama-tama data $\vec{x}$ dipetakan oleh fungsi $\Phi(\vec{x})$ ke ruang vektor yang berdimensi lebih tinggi. Pada ruang vektor yang baru ini, hyperplane yang memisahkan kedua class tersebut dapat dikonstruksikan[8]. Hal ini sejalan dengan teori Cover yang menyatakan. Jika suatu transformasi bersifat non linear dan dimensi dari feature space cukup tinggi, maka data pada input space dapat dipetakan ke feature space yang baru, dimana pattern-pattern tersebut pada probabilitas tinggi dapat dipisahkan secara linear.

Ilustrasi dari konsep ini dapat dilihat pada gambar dibawah. Pada gambar a diperlihatkan data pada class kuning dan data pada class merah yang berada pada input space berdimensi dua tidak dapat dipisahkan secara linear. Selanjutnya gambar b menunjukkan bahwa fungsi Φ memetakan tiap data pada input space tersebut ke ruang vektor baru yang berdimensi lebih tinggi (dimensi 3), dimana kedua class dapat dipisahkan secara linear oleh sebuah hyperplane. Notasi matematika dari mapping ini adalah sbb.

$$\Phi: \Re^d \longrightarrow \Re^q \;\; d < q \tag{2.15}$$



Gambar 2.15 Fungsi Φ memetakan data ke ruang vektor yang berdimensi lebih tinggi, sehingga kedua class dapat dipisahkan secara linear oleh sebuah hyperplane[7].

Table 2.1 Jenis Kernel pada SVM[8]

Jenis Kernel	Definisi
Polynomial	$K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i . \vec{x}_j + 1)^p$
Gaussian	$K(\vec{x}_i, \vec{x}_j) = \exp(\frac{\left\|\vec{x}_i - \vec{x}_j\right\|^2}{2\sigma^2})$
Sigmoid	$K(\vec{x}_i, \vec{x}_j) = \tanh(a\vec{x}_i . \vec{x}_j + \beta)$

Pemetaan ini dilakukan dengan menjaga topologi data, dalam artian dua data yang berjarak dekat pada input space akan berjarak dekat juga pada feature space, sebaliknya dua data yang berjarak jauh pada input space akan juga berjarak jauh pada feature space.

2.4.3 Karakteristik SVM

Karakteristik SVM sebagaimana telah dijelaskan pada bagian sebelumnya, dirangkumkan sebagai berikut:

1. Secara prinsip SVM adalah linear classifier

2. Pattern recognition dilakukan dengan mentransformasikan data pada input space ke ruang yang berdimensi lebih tinggi, dan optimisasi dilakukan pada ruang vector yang baru tersebut. Hal ini membedakan SVM dari solusi pattern recognition pada umumnya, yang melakukan optimisasi parameter pada ruang hasil transformasi yang berdimensi lebih rendah daripada dimensi input space.
3. Menerapkan strategi Structural Risk Minimization(SRM)
4. Prinsip kerja SVM pada dasarnya hanya mampu menangani klasifikasi dua class.

2.5 Android NDK

The NDK adalah sebuah toolset yang memungkinkan Anda untuk melaksanakan bagian dari aplikasi Anda menggunakan bahasa asli-kode seperti C dan C + +. Untuk beberapa jenis aplikasi, hal ini dapat membantu sehingga Anda dapat menggunakan kembali kode perpustakaan yang ada ditulis dalam bahasa-bahasa ini, tapi kebanyakan aplikasi tidak perlu NDK Android. Sebelum men-download NDK, Anda harus memahami bahwa NDK tidak akan menguntungkan sebagian besar aplikasi. Sebagai pengembang, Anda perlu menyeimbangkan manfaat terhadap kekurangan. Khususnya, menggunakan kode asli pada Android umumnya tidak menghasilkan peningkatan kinerja terlihat, tetapi selalu meningkatkan kompleksitas aplikasi Anda. Secara umum, Anda hanya harus menggunakan NDK jika sangat penting untuk Anda app-tidak pernah karena Anda hanya memilih untuk program di C / C + +. Calon khas baik untuk NDK yang mandiri, operasi CPU-intensif yang tidak mengalokasikan banyak memori, seperti pemrosesan sinyal, simulasi fisika, dan sebagainya. Ketika memeriksa apakah atau tidak Anda harus mengembangkan dalam kode asli, berpikir tentang kebutuhan Anda dan melihat apakah Android kerangka API menyediakan fungsionalitas yang Anda butuhkan[11].

2.6 Android Studio

Gambar 2.16 Logo Android Studio

Android Studio adalah sebuah IDE untuk Android Development diperkenalkan google pada acara I/O 2013. Android Studio menggunakan Gradle untuk memanajemen project . Gradle Merupakan Build Automation Tool, untuk mengenal lebih lanjut melalui situs berikut ini gradle.org , ini yang membedakan gradle dari Ant atau Maven yang memakai XML[12].

2.7 OpenCV

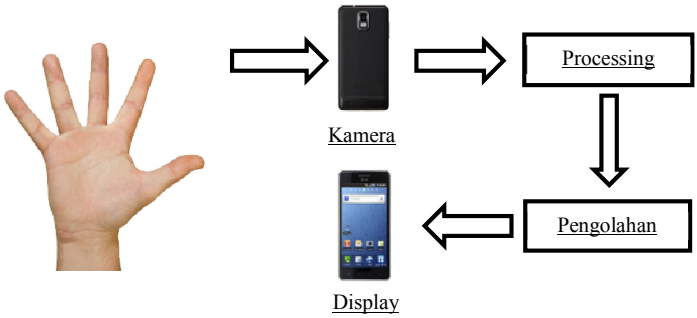
OpenCV (Open Computer Vision) adalah sebuah API (Application Programming Interface) library yang sudah sangat familiar pada pengolahan citra computer vision. Computer vision itu sendiri adalah salah satu cabang dari bidang ilmu pengolahan citra (Image Processing) yang memungkinkan komputer dapat melihat seperti manusia. Dengan computer vision tersebut komputer dapat mengambil keputusan, melakukan aksi, dan mengenali terhadap suatu objek. Beberapa pengimplementasian dari computer vision adalah face recognition, face detection, face/object tracking, road tracking, dll[13].

OpenCV adalah library open source untuk computer vision untuk C/C++ , Python, Java dan Matlab/Octave. OpenCV didesain untuk aplikasi real-time, memiliki fungsi-fungsi akuisisi yang baik untuk image/video. OpenCV juga menyediakan interface ke Integrated Performance Primitives (IPP) Intel sehingga jika anda bisa mengoptimasi aplikasi computer vision anda jika menggunakan prosesor Intel[3]. Fitur yang dimiliki OpenCV antara lain :

a. Manipulasi data citra (alocation, copying, setting, convert).
b. Citra dan video I/O (file dan kamera based input, image/video file output).
c. Manipulasi Matriks dan Vektor beserta rutin-rutin aljabar linear (products, solvers, eigenvalues, SVD).
d. Data struktur dinamis (lists, queues, sets, trees, graphs).
e. Pemroses citra fundamental (filtering, edge detection, corner detection, sampling and interpolation, color conversion, morphological operations, histograms, image pyramids).
f. Analisis struktur (connected components, contour processing, distance Transform, various moments, template matching, Hough Transform, polygonal approximation, line fitting, ellipse fitting, Delaunay triangulation).
g. Kalibrasi kamera (calibration patterns, estimasi fundamental matrix, estimasi homography, stereo correspondence).
h. Analisis gerakan (optical flow, segmentation, tracking).
i. Pengenalan obyek (eigen-methods, HMM).
j. Graphical User Interface (display image/video, penanganan keyboard dan mouse handling, scroll-bars).

BAB III
PERANCANGAN SISTEM

Pada bab ini akan dibahas mengenai perancangan pembuatan sistem perangkat lunak untuk melakukan pengolahan informasi visual. Sistem penerjemahan bahasa isyarat dengan gerakan tangan terdiri dari 4 komponen yang saling berhubungan diantaranya yaitu seorang pengguna, kamera , processing unit, pengolahan informasi visual, dan media komunikasi untuk pengguna. Hubungan antara komponen-komponen dapat di ilustrasikan pada Gambar 3.1.



Gambar 3.1 Ilustrasi cara kerja sistem

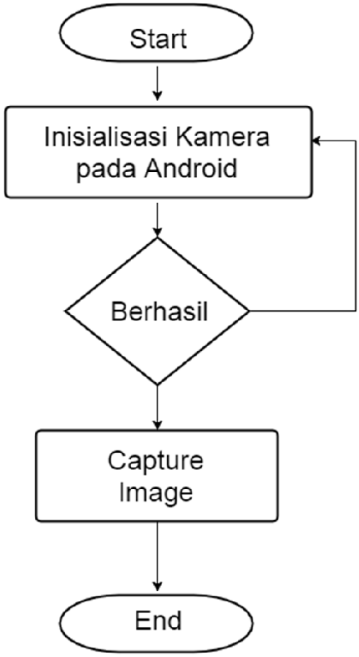
3.1 Prinsip Kerja

Menurut ilustrasi pada gambar diatas prinsip kerja dari sistem penerjemahan bahasa isyarat pada gerak tangan yang akan dirancang dimulai dari pengambilan informasi visual. Telepon genggam dengan system operasi *android* merupakan perangkat yang digunakan untuk mengambil informasi visual dari pengguna. Informasi visual yang diambil diantaranya yaitu citra RGB dan situr dari citra. Selanjutnya informasi visual tersebut akan dikirimkan menuju *processing unit*. Pada sistem yang akan dirancang, sebuah *android mobile phone* digunakan sebagai *processing unit*. Informasi visual akan diolah untuk mendapatkan estimasi gesture dari pengguna. Dan terdapat program *inteface* yang akan membantu pengguna berinteraksi dengan sistem yang akan dirancang.

3.2 Android

Android merupakan perangkat yang digunakan untuk mengakuisisi citra RGB dan gesture image, berikut ini merupakan spesifikasi dari Android Advan S50D yang digunakan:

- OS : Android Kitkat 4.4
- Memori : Internal 8GB, microSD up to 32 GB
- CPU : Quad Core 1.2 Ghz Cortex-A53
- Ram : 1 GB RAM
- Kamera Belakang : 5 MP, autofocus, LED flash
- Fitur : Geo-tagging, touch focus, face detection, panorama
- Video : 720p@30fps
- Depan : 2 MP



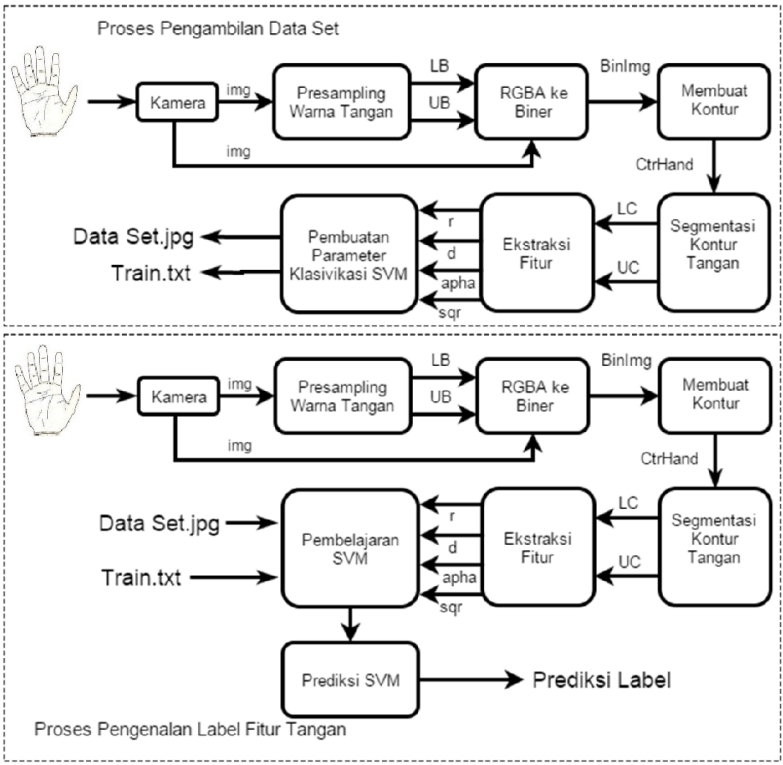
Gambar 3.2 Inisialisasi kamera pada android

Inisialisasi merupakan langkah awal dalam penggunnaan kamera pada android mobile phone. Inisialisasi ini menentukan sensor-sensor yang akan digunakan kamera untuk melakukan pengambilan data seperti yang ada pada Gambar 3.2. Disini penulis akan melakukanan inisialisasi untuk penggunaan kamera yaitu color stream, presampling color. Color stream merupakan data stream yang digunakan untuk melakukan pengambilan data citra RGB. Presampling color juga termasuk data stream yang digunakan untuk pengambilan data, tetapi data yang diambil disini yaitu berupa data warna untuk background dan warna kulit tangan yang akan digudakan untuk ekstraksi fitur.

3.3 Pengolahan Informasi Visual

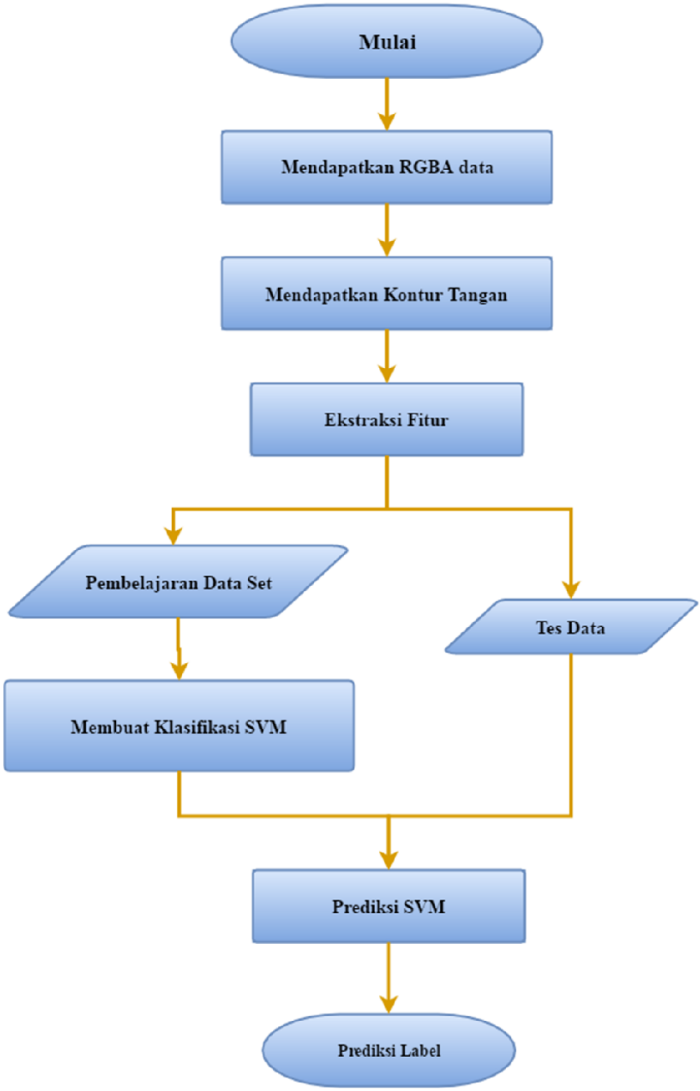
Untuk menentukan estimasi informasi fitur tangan dari penggunanya, maka sistem perlu mengambil informasi-informasi visual yang berkaitan dengan penggunanya. Langkah-langkah sistem untuk melakukan estimasi fitur tangan pengguna menurut gambar flowchart inisialisasi android adalah seperti berikut:

1. Camera menangkap informasi citra RGB, dari data RGB dilakukan pengambilan sample warna untuk untuk dijadikan batas threshold citra dan penentuan parameter fitur tangan hasil pengolahan citra.
2. Citra biner dan parameter fitur diperlihatkan ke user melalui program interface.
3. Parameter fitur memberikan informasi setiap sendi dalam koordinat hand gesture space.
4. Fitur yang diambil adalah jarak kedalam jari, radius pusat jari dan sudut pergelangan tangan.
5. Seluruh fitur yang didapat menjadi masukkan untuk jaringan saraf tiruan yang akan menentukan regresi untuk setiap masukkan yang diberikan yang pada akhirnya dapat memberikan estimasi fitur dari pengguna.
6. Pengambilan keputusan akan memberikan saran kategori label berdasar hasil estimasi fitur tangan pengguna.
7. Perkiraan label dari ekstraksi fitur yang didapatkan dari proses jaringan saraf tiruan akan disampaikan ke user melalui program interface.



Gambar 3.3 Diagram blok sistem

Gambar 3.3 menunjukkan diagram blok dari sistem. Dari gambar tersebut terlihat bahwa pemrosesan citra terbagi di dua bagian, yaitu proses pengambilan data set awal dan proses pengenalan label. Pemrosesan citra untuk data set di mulai dengan presampling warna yang nantinya dijadikan parameter untuk membuat fitur tangan pengguna yang selanjutnya akan diproses menjadi klasifikasi parameter fitur untuk media pembelajaran. Dan yang kedua yaitu proses pengenalan label, dimana dalam proses ini tidak jauh berbeda dengan proses data set hanya saja disini hasil ekstraksi fitur langsung dibandingkan dengan fitur yang ada pada data set sehingga akan didapat label yang sesuai dengan bentuk tangan pengguna.

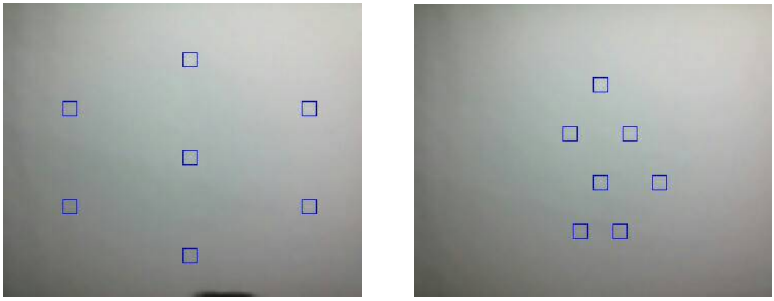


Gambar 3.4 Diagram alur sistem

Pada Gambar 3.4, diagram alur dari sistem digambarkan dari awal yaitu mulai dari mendapatkan informasi citra RGBA. Secara garis besar, prosesnya sama seperti pada blok diagram. Hanya saja, pada diagram alur, dijelaskan pula proses pembelajaran data set yang selanjutnya akan dijadikan sebuah klasifikasi untuk pembelajaran pengenalan label.

3.3.1 Pengambialn Sampel Warna

Proses pertama yaitu pengambilan sample warna. Pada proses ini terbagi menjadi dua bagian proses, yaitu pengambilan sampel warna latar dan tangan pengguna. Langkah pertama untuk visualisasi informasi latar atau background dari frame yaitu dengan cara melakukan presampling yang mengumpulkan warna latar belakang menggunakan 7 kotak kecil yang ditampilkan pada layar seperti Gambar 3.5a. Yang mana data warna ini selanjutnya digunakan untuk menghitung ambang batas untuk mendapatkan citra biner dari data masukan RGBA. Untuk mempermudah, metode adaptif tidak digunakan untuk menemukan ambang terbaik. Bagian kedua yaitu visualisasi informasi warna tangan atau kulit pengguna menggunakan 7 kotak kecil yang ditampilkan pada layar seperti halnya langkah pengambilan sampel layar. Untuk proses yang kedua ini Pengguna diminta untuk meletakkan tangannya dekat dengan layar untuk menutupi 7 kotak seperti yang ada pada Gambar 3.5b sehingga program bisa mendapatkan data warna tangan. Perhatikan bahwa jumlah kotak hanya sebuah nilai empiris dan dapat nilai-nilai lainnya. Setelah 7 data warna untuk tangan diperoleh, 7 batas atas dan bawah untuk daerah tangan dihitung, yang dapat direpresentasikan sebagai vektor 6 dimensi.



Gambar 3.5 Pengambilan 7 pattern warna. (a). pattern wana latar, (b). pattern warna tangan

Karena bagi tangan yang sama, warna biasanya bervariasi paling pada dimensi pencahayaan dan lebih kecil dari warna dan dimensi warna lain, ruang warna yang diinginkan harus dapat memisahkan mereka.Bounding vector ditentukan oleh pengembang dan tidak bisa diubah. Oleh karena itu, kinerja segmentasi sebenarnya tergantung pada pilihan vektor bounding. Untuk lebih memahami pengaruh setiap elemen dari vektor pada kinerja segmentasi, ruang warna RGB asli diubah ke banyak ruang warna lain seperti HSV, YCrCb, CIE L * a * b *.

3.3.2 Proses Pengambilan Kontur Terbesar hasil Tresholding

Ketika presampling telah dilakukan maka akan dilakukan pemilihan kontur yang paling besar pada hasil pengambilan gambar. Pemilihan kontur terbesar ini difungsikan agar error hasil threshold bisa dikurangi sehingga yang didapat hanya gambar tangan saja. Pada proses threshold ini terdapat tahapan agar gambar tangan yang didapatkan bisa menyerupai bentung tangan manusia secara utuh.

Tahap pertama yaitu proses penghalusan kontur. Dimana pada tahap ini citra akan dilakukan dilatasi dan erosi secara bersamaan agar kontur yang didapat mendekati aslinya. Pada tahap ini dilakukan dilatasi sebanyak 3 kali dan erosi sebanyak 2 kali.

Tahap kedua yaitu proses penghalusan (smoothing) citra gambar hasil erosi dan dilatasi. Karena yang dilakukan proses penghalusan citra aalah citra biner maka proses ini hanya dilakukan dalam satu channel warna saja. Sehingga proses ini dilakukan dengan perhitungan seperti pada persamaan 3.1.

$$\bar{\mathbf{c}}(x, y) = \frac{1}{K} \sum_{(s,t) \in S_{xy}} \mathbf{c}(s, t) \tag{3.1}$$

Gambar 3.6 Hasil Pemilihan kontur terbesar pada layar

Setelah kedua proses diatas selesai maka baru dilakukan proses pemilihan kontur terbesar. Pemilihan kontur terbesar ini digunakan untuk menghilangkan kontur-kontur kecil yang berada di luar kontur tangan dan masih berada pada batasan threshold. Karena jika kontur-kontur kecil ini dibiarkan maka akan terjadi kesalahan pembacaan data untuk proses selanjutnya. Sehingga kontur yang di dapatkan hanyalah kontur tangan saja seperti pada Gambar 3.6. Berikut adalah list program untuk mendapatkan kontur terbesar .

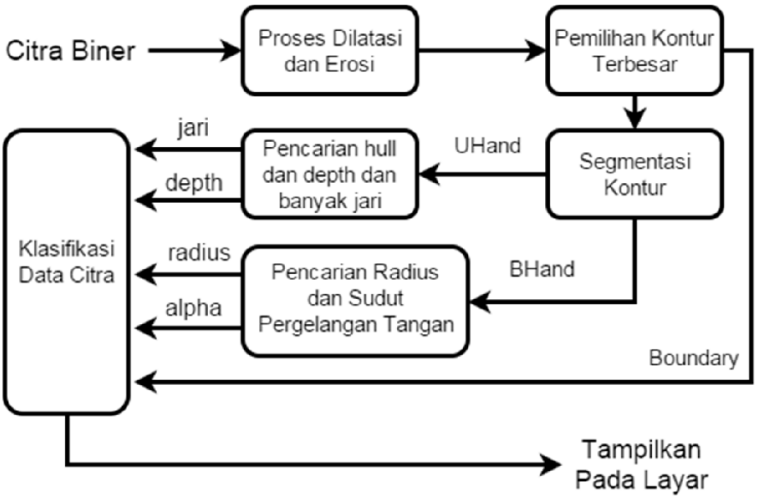
```
void findBiggestContour()
{
   int idx = -1;
   int cNum = 0;
   for (int i = 0; i < contours.size(); i++)
   {
      int curNum = contours.get(i).toList().size();
      if (curNum > cNum) {
         idx = i;
         cNum = curNum;
      }
   }
   cMaxId = idx;
}
```

3.3.3 Pengambilan Fitur Citra

Fitur yang digunakan untuk mewakili tangan adalah fitur tingkat rendah, yang pada dasarnya adalah vektor jari. Untuk menghitung mereka, perlu untuk mengetahui lokasi dari pusat telapak tangan dan ujung jari. Beberapa fungsi OpenCV dapat langsung mengembalikan kontur dan convex hull poin dari tangan dari gambar biner yang berisi sisi tersegmentasi. Menggunakan koordinat pusat dan square tracking, pusat telapak tangan dan jari-jari lingkaran tertulis dari kontur dapat dihitung. Bagian ini cukup komputasi mahal dan diimplementasikan dalam native C ++ kode.

Untuk tahap ini dilakukan langkah-langkah sebagai berikut seperti yang ada pada digram blok pada Gambar 3.7 :

1. Mengaktifkan sensor untuk mengangkap objek dengan menggunakan kamera yang terdapat pada android.
2. Capture image dilakukan secara real time memakai kamera pada tampilan aplikasi yang telah dibuat.



Gambar 3.7 Blok diagram pengambilan fitur

3. Setelah itu dilakukan deteksi objek tangan. Pada proses ini untuk mendeteksi objek tangan saja dan supaya membedakan dengan objek lainnya digunakanlah presampling pattern warna.
4. Kemudian langkah yang harus dilakukan selanjutnya segmentasi objek tangan menjadi *upperHand* dan *bottonHand* untuk memudahkan ekstraksi gesture.
5. Image tangan yang telah disegmentasi dikonversikan kedalam parameter-parameter yang telah ditentukan. Yang selanjutnya parameter-parameter ini akan ditampilkan pada layar interface yang dirancang.

Gambar 3.7 menunjukkan ada 5 parameter yang akan dicari dalam proses pendeteksian fitur. Dimana ke-lima parameter fitur dicari dengan algoritma sebagai berikut:

- Pencarian titik pusat tangan

  Proses pencarian titik pusat ini adalah dengan mencari distribusi penyebaran titik-titik pixel pada hasil threshold kontur tangan yang telah didapat pada persamaan 3.3.

$$\bar{x} = \frac{\sum_{i=0}^{k} x_i}{k}, \frac{\sum_{i=0}^{k} y_i}{k} \tag{3.2}$$

Dimana x_i dan y_i adalah kordinat dari x, y dari i pixel pada area tangan serta k menunjukkan jumlah pixel dalam area tangan tersebut.

```
if (fid != 0) {
     Point prevFinPoint = finTipsTemp.get(fid-1);
     curFinPoint.x = (curFinPoint.x + prevFinPoint.x)/2;
     curFinPoint.y = (curFinPoint.y + prevFinPoint.y)/2;
        }
```

- Mencari radius telapak tangan dari pusat tangan

  Untuk mendapatkan radius dari telapak tangan ini adalah dengan memanfaat proses sebelumya ketika titik pusat tangan telah didapatkan maka yang selanjutnya adalah dengan menarik garis horizontal dari pusat tangan hingga mendapatkan pixel hitam pertama kali setelah itu diterapkan jarak eucliden untuk mendapatkan radius dari telapak tangan tersebut [3.4].

$$d(p,q) = \sqrt{(p_1 - q_1)^2 - (p_1 - q_1)^2 - (p_1 - q_1)^2} \quad (3.3)$$

Berikut adalah list program untuk mendapatkan radius telapak tangan:

```
Point disFinger = new Point(curFinPoint.x-inCircle.x,
curFinPoint.y-inCircle.y);
double dis =
Math.sqrt(disFinger.x*disFinger.x+disFinger.y*disFinger.y);
Double f1 = (disFinger.x)/inCircleRadius;
Double f2 = (disFinger.y)/inCircleRadius;
features.add(f1);
features.add(f2);
```

- Mencari boundary

  Untuk pencarian boundary ini memakai algoritma convex hull yaitu dengan mencari subset dari himpunan titik pada bidang tangan sehingga jika titik-titik terluar dari bagian tangan ini disatukan menjadi polygon maka akan menjadi polygon yang konveks. Dengan menggunakan persamaan 3.5.

$$\left\{\sum_{i=1}^{|S|} \alpha_i x_i \,\middle|\, (\forall_i : \alpha_i \geq 0) \wedge \sum_{i=1}^{|S|} \alpha_i = 1\right\} \quad (3.4)$$

Berikut adalah list program untuk mendapatkan boundary telapak tangan:

```
boolean detectIsHand(Mat img){
                int centerX = 0;
                int centerY = 0;
                if (boundingRect != null) {
                        centerX = boundingRect.x +
boundingRect.width/2;
                        centerY = boundingRect.y +
boundingRect.height/2;
                }
                if (cMaxId == -1)
                        isHand = false;
                else if (boundingRect == null) {
                        isHand = false;
                } else if ((boundingRect.height == 0) ||
(boundingRect.width == 0))
                        isHand = false;
                else if ((centerX < img.cols()/4) ||
(centerX > img.cols()*3/4))
                        isHand = false;
                else
                        isHand = true;
                return isHand;
        }
```

- Mencari sudut pada pergelangan tangan

  Untuk mencari sudut pergelangan tangan dimudahkan karena sudut ini sudah terekstrasi pada proses convex hull sehingga kita hanya perlu mengaksesnya dengan fungsi prevDefectVec pada OpenCV untuk mendapatkan selisih sudut dari 2 vektor hasil convex hull.

Berikut adalah list program untuk mendapatkan titik konvexs telapak tangan:

```
if ((detectIsHand(img))) {
defectMat.fromList(defectPoints);
List<Integer> dList = defects.toList();
Point[] contourPts = contours.get(cMaxId).toArray();
Point prevDefectVec = null;
        int i;
        for (i = 0; i < defectIdAfter.size(); i++){
```

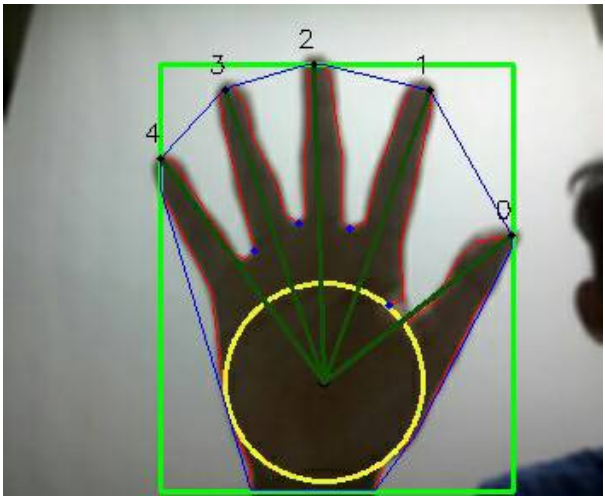
```
        int curDlistId = defectIdAfter.get(i);
        int curId = dList.get(curDlistId);
        Point curDefectPoint = contourPts[curId];
        Point curDefectVec = new Point();
        curDefectVec.x = curDefectPoint.x - inCircle.x;
        curDefectVec.y = curDefectPoint.y - inCircle.y;

        if (prevDefectVec != null) {
        double dotProduct = curDefectVec.x*prevDefectVec.x
+       curDefectVec.y*prevDefectVec.y;
        double crossProduct =
curDefectVec.x*prevDefectVec.y -
        prevDefectVec.x*curDefectVec.y;

        if (crossProduct <= 0)
                break;
        }
        prevDefectVec = curDefectVec;

        }

        int startId = i;
        int countId = 0;
```



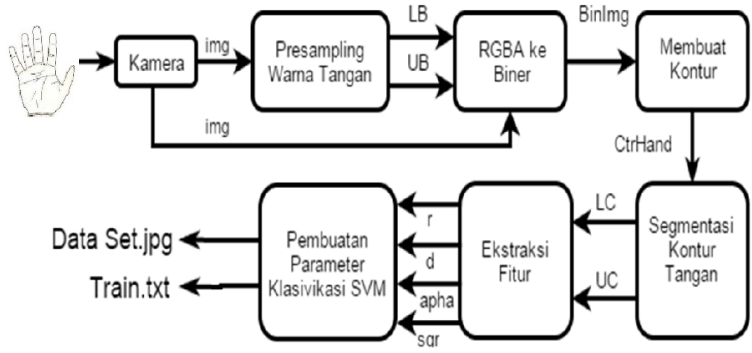
Gambar 3.8 Hasil pendektesian gesture tangan

Lokasi dari ujung jari dihitung dengan menggunakan convexcity defect. Untuk melakukannya, convexcity defect berlebih dieliminasi dengan memeriksa kendala pada kedalaman dan sudut dari titik convex dan sebagainya, seperti yang ditunjukkan pada ambar 3.8. Berikutnya convexcity defect meninggalkan yang mengatur kembali dan ujung jari yang diperoleh dari koordinat kembali fungsi OpenCV. Akhirnya vektor jari dihitung dan dibagi dengan radius lingkaran Inscribe untuk mendapatkan vektor fitur akhir:

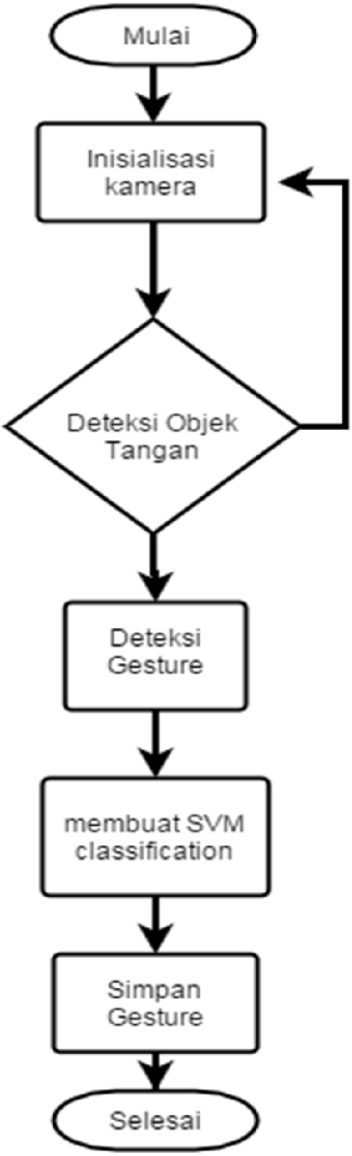
$$X = [x_0/r\,, y_0/r\,, x_1/r\,, y_1/r \ldots x_4/r\,, y_4/r] \tag{3.5}$$

3.3.4 Proses Pengambilan Data Set Awal

Pada proses pengambilan data jumlah data image yang diambil untuk setiap bentuk fitur tangan diambil sebanyak 24 buah ,dimana image ini merepresentasikan huruf A-Z (kecuali huruf J dan Z). Dari setiap hasil capture image deteksi objek disimpan dalam chace system dan di backup dalam bentuk image rgb dengan extention .jpg dengan lebar pixel 480p. Pada Gambar 3.9 diketahui ada 4 parameter yag akan diklasifikasikan untuk pembelajaran svm yaitu radius (r) , kedalaman jari (d), sudut pergelangan tangan (alpha), boundary square (sqr). Klasifikasi disini adalah penyederhanaan parameter yang telah ditentukan kedalam fungsi-fungsi dalam ruang vektor. Yang mana fungsi dalam ruang vektor inilah yang menjadi basis metode pembelajaran SVM.



Gambar 3.9 Blok diagram pengambilan data set



Gambar 3.10 Diagram alir penyimpanan data image

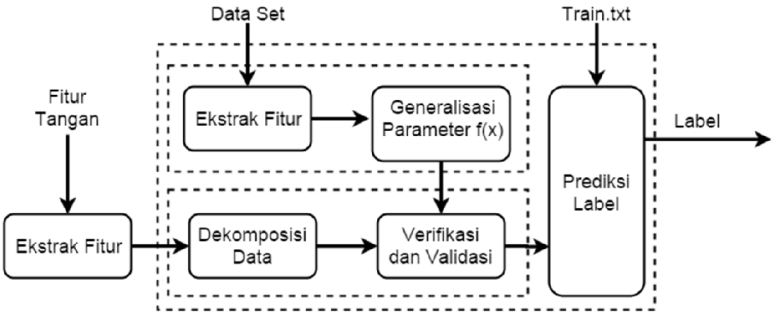
Gambar 3.9 menjelaskan alur proses penyimpanan fitur tangan. Dimana merukapan representasi sederhana blok diagram pada Gambar 3.8. Proses penyimpanan fitur ini bisa dilakukan secara otomatis dengan memanfaatkan tombot add gesture pada system yang dibuat. Dimana ketika pada layar interface terlihat system telah mendapatkan gesture maka pengguna menekan tombol add gesture pada layar lalu menunggu 10 detik untuk membuat svm classification selanjutnya data set akan tergenerate otomatis kedalam system dan di backup kedalam bentuk image.

3.3.5 Support Vector Machine

Support Vector Machine (SVM) dalam dunia machine learning adalah pembelajaran yang menggunakan ruang hipotesis berupa fungsi-fungsi linier dalam sebuah ruang fitur (feature space) berdimensi tinggi, dilatih dengan algoritma pembelajaran yang didasarkan pada teori optimasi dengan mengimplementasikan learning bias yang berasal dari teori pembelajaran statistik. SVM dapat melakukan mapping antara masukkan dan keluaran ataupun mencari korelasi antar keduanya. Korelasi didapatkan dari pasangan masukkan dan keluaran yang telah dilatihkan.

Penggunaan SVM karena kemampuannya untuk melakukan generalisasi pada keluaran, meskipun informasi-informasi yang diterima tidak ada pada informasi-informasi yang telah dilatihkan. Selain itu SVM juga dapat menentukan tren data yang sangat kompleks yang tidak terlihat oleh pengamatan manusia.

SVM sangat membantu bila pada suatu sistem membutuhkan multiple input dan juga multiple output. Bila sistem menggunakan single input dan single output maka metode one-against-one sudah cukup untuk di implementasikan pada sistem. Pada Tugas Akhir ini SVM digunakan untuk melakukan pembelajaran agar dapat memperkirakan label dari perubahan gerak tangan manusia. Pembelajaran SVM menggunakan library OpenCV. Penulis menggunakan OpenCV untuk mempercepat proses pembelajaran dan menghindari overfit yang dapat terjadi. Structural Risk Minimization (SRM) digunakan untuk untuk menjamin batas atas dari generalisasi pada data pengujian dengan cara mengontrol ”kapasitas” (fleksibilitas) dari hipotesis hasil pembelajaran.



Gambar 3.11 Diagram blok SVM

Gambar 3.10 menunjukkan proses pembelajaran SVM yang di rancang. Dimana setelah pengambilan fitur tangan akan dilakukan dekomposisi data yang bertujuan untuk normalisasi masukan parameter yang selanjutnya akan divalidasi dengan parameter fitur pada data set yang telah di generalisasi fungsinya.. SVM yang diterapkan disini yaitu metode one-against-all. Dimana setiap model klasifikasi ke-i dilatih dengan menggunakan keseluruhan data, untuk mencari solusi permasalahan. Fungsi generalisasi yang digunakan yaitu kernel sigmoid. Fungsi generalisasi yang digunakan bertujuan untuk meningkatkan performansi dalam mendapatkan tren data dari perubahah gesture akibat perubahan gerakan tangan. Adapun masukkan yang diberikan yaitu:

- Jarak kedalaman (depth) antara pangkal jari (titik biru) dengan boundary(garis biru).
- Cosines Sudut pergelangan tangan.
- Radius pusat telapak tangan.
- Garis boundary secara keseluruhan (garis biru dan kotak hijau).

Setelah sistem selesai melakukan pembelajaran maka nilai tiap parameter di eksport kedalam file jpeg beserta gambar gesture dan siap digunakan untuk tahapan estimasi pengenalan perubahan pergerakan tangan. pengenalan perubahan pergerakan tangan atau bagian one-against-all di terapkan pada C++ dimulai dengan import data-data jpeg sebagai data set awal yang selanjutnya dilakukan pembelajaran keseluruhan dengan nilai inputan yang diterima.

Algoritma SVM yang penulis gunakan dalam tugas akhir ini adalah algoritma dekomposisi dengan working set. prinsip dekomposisi adalah mengoptimasi masalah global dengan hanya menggunakan

sebagian kecil data pada satu saat. Sehingga data yang masuk pada pelatihan hanya sebagian kecil.

Teknik dekomposisi secara matematis dapat direpresentasikan dalam notasi matriks. Dimisalkan 4 parameter diatas sebagai berikut dimana $\alpha = (\alpha_1, \ldots \alpha_l)^T, y = (y_1, \ldots y_l)^T,\ \ Q_{ij} = y_i y_j K(x_i, x_j)$, dan e merupakan vektor dengan jumlah elemen sebanyak l (jumlah data pelatihan) dan semuanya bernilai 1. Maka SVM dual problem dapat dituliskan sebagai berikut:

$$\begin{aligned} &\max_{\alpha} e^T\alpha - \frac{1}{2}\alpha^T Q\alpha \\ &s.t. 0 \le \alpha_i \le C, i = 1 \ldots l \\ &y^T\alpha = 0 \end{aligned} \tag{3.6}$$

Sehingga vektor α dapat dibagi menjadi α_B untuk dimasukkan kedalam pelatihan dan α_N adalah variable sisanya. Selanjutnya matriks Q dapat dipartisi menjadi:

$$Q = \begin{bmatrix} Q_{BB} & Q_{BN} \\ Q_{NB} & Q_{NN} \end{bmatrix} \tag{3.7}$$

dimana setiap bagiannya ditentukan oleh himpunan indeks B dan N.

Selanjutnya dibuat kesimpulan awal jika α^k adalah titik stasioner dari persamaan diatas maka proses berhenti dan jika tidak maka ditentukan $B = \{i, j\}$.

Jika $\alpha_{ij} \equiv K_{ij} + K_{jj} - 2K_{ij} > 0$

$$\begin{aligned} &\min_{\alpha_i, \alpha_j} \frac{1}{2}\begin{bmatrix} \alpha_B^T & (\alpha_N^k)^T \end{bmatrix}\begin{bmatrix} Q_{BB} & Q_{BN} \\ Q_{NB} & Q_{NN} \end{bmatrix}\begin{bmatrix} \alpha_B \\ \alpha_N^k \end{bmatrix} - \begin{bmatrix} e_B^T & e_N^T \end{bmatrix}\begin{bmatrix} \alpha_B \\ \alpha_N^k \end{bmatrix} \\ &= \frac{1}{2}\alpha_B^T Q_{BB}\alpha_B + (-e_B + Q_{BN}\alpha_N^k)^T\alpha_B + kons\tan \\ &= \frac{1}{2}[\alpha_i\ \alpha_j]\begin{bmatrix} Q_{ii} & Q_{ij} \\ Q_{ij} & Q_{jj} \end{bmatrix}\begin{bmatrix} \alpha_i \\ \alpha_j \end{bmatrix} + \left(-e_B + Q_{BN}\alpha_N^k\right)^T\begin{bmatrix} \alpha_i \\ \alpha_j \end{bmatrix} + kons\tan \\ &s.t. 0 \le \alpha_t \le C, t \in B \\ &y_B^T\alpha_B^T = -y_N^T y_N^k \end{aligned} \tag{3.8}$$

Jika tidak , maka sistem akan diselesaikan dengan

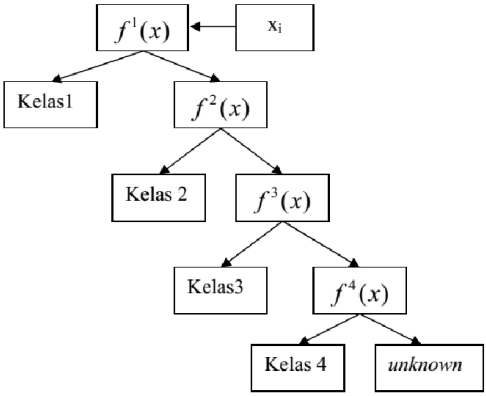
$$\min_{\alpha_i,\alpha_j} \frac{1}{2}[\alpha_i \ \alpha_j] \begin{bmatrix} Q_{ii} & Q_{ij} \\ Q_{ij} & Q_{jj} \end{bmatrix} \begin{bmatrix} \alpha_i \\ \alpha_j \end{bmatrix} + \left(-e_B + Q_{BN}\alpha_N^k\right)^T \begin{bmatrix} \alpha_i \\ \alpha_j \end{bmatrix} + \frac{\tau - a_{ij}}{4}\left(\left(\alpha_i - {}_i^k\right)^2 + \left(\alpha_i - {}_j^k\right)^2\right) \tag{3.9}$$

Selanjutnya α_B^{k+1} di set sebagai solusi optimal dari subproblem diatas sehingga $\alpha_N^{k+1} \equiv \alpha_N^k$

Karena syarat terjadinya titik stasioner adalah jika $m(\alpha) \leq M(\alpha)$ yang merupakan kondisi Karush-Kuhn-Tucker (KKT). Maka penyeselaian akhirnya adalah:

$$\begin{aligned} m(\alpha) &\equiv \max_{i \in I_{up}(\alpha)} -y_i \nabla f(\alpha)_i \\ M(\alpha) &\equiv \min_{i \in I_{low}(\alpha)} -y_i \nabla f(\alpha)_i \\ I_{up}(\alpha) &\equiv \left\{ t \mid \alpha_t < C, y_t = 1 \ atau \ \alpha_t > 0, y_t = -1 \right\} \\ I_{low}(\alpha) &\equiv \left\{ t \mid \alpha_t < C, y_t = -1 \ atau \ \alpha_t > 0, y_t = 1 \right\} \\ \nabla f(x) &\equiv Q\alpha + e \end{aligned} \tag{3.10}$$

Lalu setelah fungsi dekomposisi didapatkan selanjutnya mnentukan jenis SVM yang digunakan. Pada tugas akhir ini penulis menggunakan jenis one-against-all yaitu klasifikasi ke-i dilatih dengan menggunakankeseluruhan data, untuk mencari solusi permasalahan.



Gambar 3.12 Ilustrasi metode one-against-all

Dengan menggunakan one-against-all hasil dekomposisi dapat disederhanakan lagi menjadi persamaan 3.11 seperti berikut:

$$F(i) \equiv \frac{\left(\bar{x}_i^{(+)} - \bar{x}_i\right)^2 + \left(\bar{x}_i^{(-)} - \bar{x}_i\right)^2}{\frac{1}{n_+ - 1}\sum_{k=1}^{n_+}\left(x_{k,i}^{(+)} - \bar{x}_i^{(+)}\right)^2 + \frac{1}{n_- - 1}\sum_{k=1}^{n_-}\left(x_{k,i}^{(-)} - \bar{x}_i^{(-)}\right)^2} \quad (3.11)$$

dimana $\bar{x}_i, \bar{x}_i^{(+)}, \bar{x}_i^{(-)}$ adalah rata-rata dari nilai atribut keseluruhan data, data positif dan data negatif. $x_{k,i}^{(+)}$ adalah fitur ke-i dari data kelas positif ke-k dan sebaliknya $x_{k,i}^{(-)}$ adalah fitur ke-i dari date kelas negatif ke-k. Semakin besar nilai F(i), maka atribut ini dapat dianggap lebih diskriminatif (lebih penting).

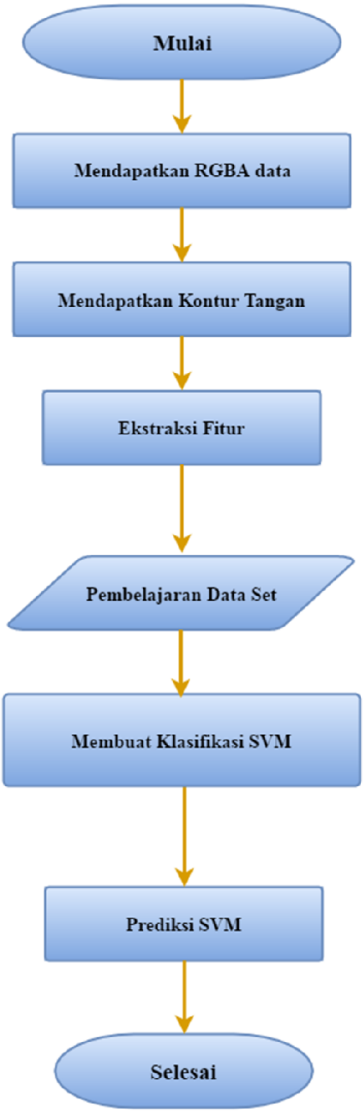
Berikut adalah list program untuk mengkonversi fitur ditunjukkan oleh label ke bentuk string yang digunakan dalam SVM input file:

```
String feature2SVMString(int label){
                String ret = Integer.toString(label) + " ";
                int i;
                for (i = 0; i < features.size(); i++)
                {
                int id = i + 1;
                ret = ret + id + ":" + features.get(i) + " ";
                }
                ret = ret + "\n";
                return ret;
        }
```

Berikut adalah list program untuk training SVM dengan mengakses LibSVM pada native OpenCV.

```
     private native int trainClassifierNative(String
trainingFile, int kernelType,
                int cost, float gamma, int isProb, String
modelFile);
    private native int doClassificationNative(float values[][],
int indices[][],
                int isProb, String modelFile, int labels[],
double probs[]);
    private void train() {
        // Svm training
```

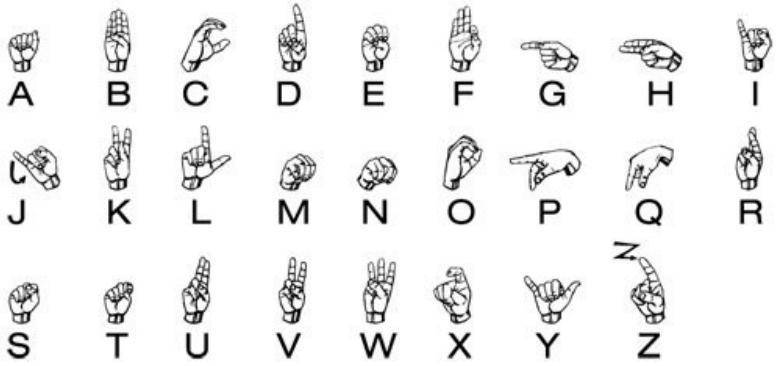
```
        int kernelType = 2; // Radial basis function
        int cost = 4; // Cost
        int isProb = 0;
        float gamma = 0.001f; // Gamma
        String trainingFileLoc = storeFolderName+DATASET_NAME;
        String modelFileLoc = storeFolderName+"/model";
        Log.i("Store Path", modelFileLoc);
        if (trainClassifierNative(trainingFileLoc, kernelType,
cost, gamma, isProb,
    modelFileLoc) == -1) {
                Log.d(TAG, "training err");
                finish();
        }
        Toast.makeText(this, "Training is done", 2000).show();
    }
        public void initLabel() {
                File file[] = storeFolder.listFiles();
                int maxLabel = 0;
                for (int i=0; i < file.length; i++){
                        String fullName = file[i].getName();
                final int dotId = fullName.lastIndexOf('.');
        if (dotId > 0) {
                String name = fullName.substring(0, dotId);
                String extName = fullName.substring(dotId+1);
                                if (extName.equals("jpg")) {
                        int curName = Integer.valueOf(name);
                        if (curName > maxLabel)
                        maxLabel = curName;
                                }
                        }
                }
                curLabel = maxLabel;
                curMaxLabel = curLabel;
        }
```



Gambar 3.13 Diagram alir pembelajaran SVM

3.3.6 Pengambilan Keputusan

Estimasi perubahan gerak tangan dilakukan dengan mengambil fitur-fitur dari tangan pengguna yang berada di depan kamera yang dilanjutkan dengan proses SVM untuk pengenalan perubahan gerak tangan. Proses pertama adalah memasukkan 6 file gesture sebagai data awal selanjutnya dengan pengambilan fitur yang selanjutnya akan dilakukan generalisasi untuk mendapatkan kategori label berdasarkan estimasi kedalaman (depth) pangkal jari dan lebar sudut hasil pengambilan fitur. Dimana pergerakan bentuk tangan di kategorikan kedalam huruf-huruf sebagai seperti pada Gambar 3.13.

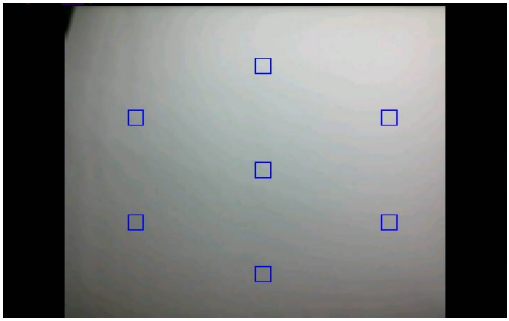


Gambar 3.14 Kategori bentuk tangan untuk merepresentasikan bentuk huruf.

3.3.7 Interface Program

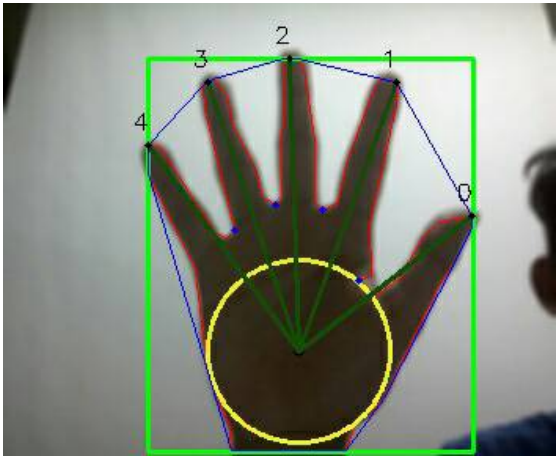
Program interface merupakan sebuah antarmuka antara sistem yang dirancang dengan pengguna. Melalui interface ini, pengguna dapat berinteraksi dengan sistem dan melihat tampilan dirinya disertai dengan hand tracking. Untuk kepentingan penerjemahan sistem dilengkapi dengan mode training dan mode test untuk mempermudah pengguna dalam pembelajaran menggunakan aplikasi ini.

Interface awal pada system ini adalah tampilan langsung untuk pengambilan pattern untuk latar dan warna kulit yang terdiri dari 7 pattern warna seperti Gambar 3.15. Dimana penggunan diharusankan melakukan sentuhan pada layar untuk mendapatkan sampel warna dari latar dan kulit tangan. Setelah pengambilan sampling selesai maka hasil capture akan dijadikan gambar shiluette atau citra biner seperti Gambar 3.16.



Gambar 3.15 Interface Awal Sistem

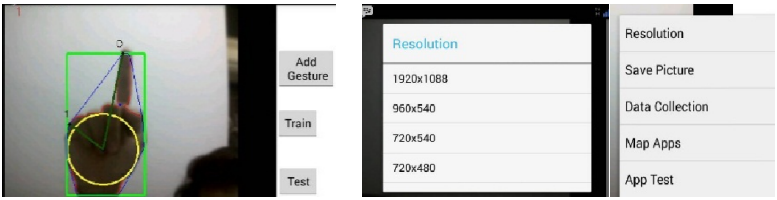
Gambar 3.16 biner image hasil pengambilan sampling



Gambar 3.17 Hasil Ekstrak Fitur

Dari hasil gambar biner pengguna harus melakukan satu sentuhan lagi ke layar untuk mendapatkan fitur tangan yang diinginkan. Hasil fitur yang didapat seperti pada Gambar 3.17 berupa tracking tangan yang nantinya akan mengikuti kemana tangan akan bergerak. Serta hasil fitur menampilkan data vektor yang nantinya akan dijadikan sebagai data untuk memprediksi huruf yang diimplementasikan oleh perubahan gerak tangan.

Setelah hasil gesture didapatkan maka tampilan interface akan berubah. Dimana aka nada tambahan 3 button touch pada layar yaitu add gesture, train dan test seperti pada Gambar 3.18a. Add gesture digunakan untuk menambahkan learning set untuk metode pembelajaran yang disediakan. Add gesture ini ada agar pengguna dapat menambahkan gesture sesuai dengan keinginan mereka. Train disini digunakan agar bisa mengetahui apakah system bisa mengenali gesture-gesture yang telah kita tambahkan. Selanjutnya adalah test yang digunakan untuk menjalankan system secara keseluruhan.



Gambar 3.18 Interface akhir system (a). Tampilan layar terakhir, (b). Menu tambahan untuk pengguna.

Interface tambahan merupakan menu yang sengaja ditambahkan untuk mempercepat kinerja sistem serta untuk mempermudah pengguna untuk menggunakan sistem yang telah dirancang. Seperti yang terlihat pada Gambar 3.18b terdapat 3 menu tambahan yaitu resolusi, simpan gambar, dan koleksi data. Resolusi ini digunakan untuk meningkatkan atau menurunkan kualitas citra. Tujuan utama dari resolusi ini adalah untuk menyesuaikan pengambilan citra sesuai dengan proscessor android yang dimiliki pengguna sehingga pengguna dapat mendapatkan kemampuan optimum pengambilan fitur tangan. Lalu menu simpan gambar ini digunakan untuk pengambilan data pengujian sehingga bila ada sebuah gerakan yang mengalami kesalahan pembacaan label, gerakan tersebut bisa dijadikan parameter unutk pembenahan sistem kedepannya. Dan yang terakhir adalah koleksi data yang mana menu ini adalah untuk digunakan pengguna dalam memvalidasi semua data set apakah sudah masuk ke sistem ataukah belum. Dimana data set ini berupa kumpulan file dengan ekstensi .jpeg yang disimpan pada folder khusus pada penyimpanan berkas di telepon genggam android.

Halaman ini sengaja dikosongkan

BAB IV
PENGUJIAN DAN ANALISIS

Pengujian sistem dimaksudkan untuk mengetahui hasil dari sistem yang telah dirancang. Pengujian ini dimulai dengan pengukuran jarak tangan pengguna terhadap layar, pengkondisian cahaya, background yang berbeda, dan penentuan label tiap perubahan gerak tangan.

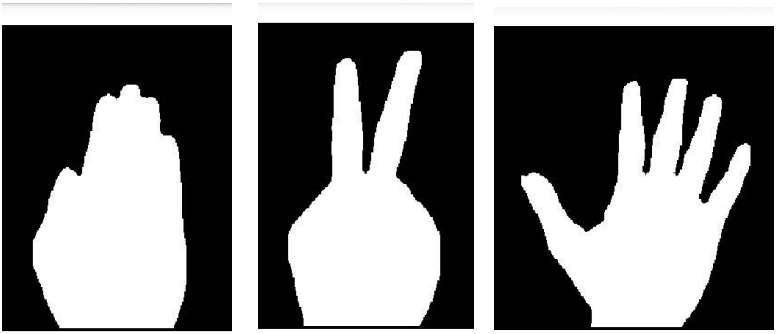
4.1 Kondisi Ideal Pengujian

Sistem ini akan bekerja lebih baik ketika berada pada kondisi ideal dimana dapat mendeteksi perbedaan warna latar dan warna tangan secara baik sehingga pembacaan gesture bisa dilakukan secara maksimal. Berikut adalah kondisi-kondisi yang harus terpenuhi untuk pengujian dalam kondisi ideal sistem:

- Didepan kamera hanya terdapat satu orang.
- Pencahayaan normal tidak terlalu terang dan terlalu gelap.
- Kondisi latar hanya terdiri dalam satu warna.
- Kondisi latar tidak boleh mengkilat atau berkaca.
- Pencahayaan tidak boleh dalam kondisi backlight.

4.2 Pengujian Contour Convexity

Countour convexity adalah metode yang gunakan untuk mencari kontur terbesar dalam layar. Kenapa kontur terbesar adalah karena pada proses presampling pengambilan citra pada proses tresholding akan mengalami cacat sehingga metode ini di gabung dengan metode morfologi filter, yaitu filter dilatasi dan erotasi untuk mendapatkan kontur yang mendekati bentuk aslinya.

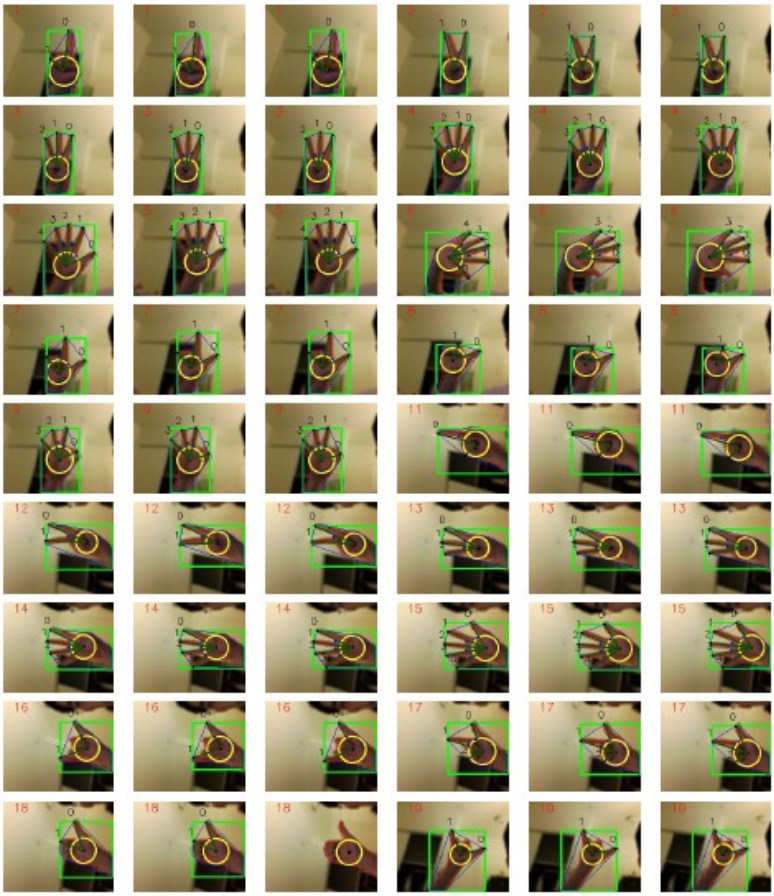


Gambar 4.1 Hasil Contour Convexity dan filter Dilatasi dan Erosi

Dari hasil Gambar 4.1 diatas dapat kita lihat metode Countour convexity bekerja dengan baik. Dimana system dapat mensegmentasi kontur sesuai bentuk tangan pengguna. Dari ketiga gambar diatas pula bisa dilihat bahwa filter dengan erotasi 3 kali dan dilatasi 2 kali hasil Countour convexity lebih halus dimana terlihat segmentasi jari jari bisa terbagi secara sempurna

4.3 Pengujian Akurasi Deteksi Terhadap Cahaya

Untuk pengujian ini sistem yang telah dibuat diujicobakan pada kondisi pencahayaan yang berbeda-beda, yaitu pada kondisi gelap, normal (terang), remang-remang, terlalu terang. Pengujian ini bertujuan sejauh mana bentuk objek tangan dengan metode contour convexity dapat di deteksi Gambar 4.2.



Gambar 4.2 Pengujian akurasi terhadap cahaya

Tabel 4.1 Pengujian akurasi terhadap cahaya

Uji ke-	Kondisi Cahaya			
	Gelap	Normal	Redup	Terang
1	X	V	V	X
2	X	V	V	V
3	X	V	V	V
4	X	V	V	V
5	X	V	V	X
6	X	V	V	X
7	X	V	X	V
8	X	V	V	V
9	X	V	V	V
10	X	V	V	V
Tingkat Akurasi	0%	100%	90%	70%

Dari hasi percobaan pada Tabel 4.1 dapat dilihat bahwa faktor pencahayaan sangat mempengaruhi deteksi objek tangan. Sehingga dalam hal ini system yang dijalankan harus berada pada kondisi pencahayaan dimana kamera dapat mendapatkan kontur tangan secara jelas. Seperti pada data diatas pencahayaan yang redup tidak terlalu berpengaruh pada pendektian kontur tangan. Namun ketika pencahayaan di terangkan maka yang terjadi adalah efek kekuningan pada tepi jari jari sehingga mengganggu proses contour convexity untuk mendapatkan kontur tangan secara tepat.

4.4 Pengukuran Jarak Tangan

Pengujian jarak ini dilakukan dengan cara mendekatkan dan menjauhkan tangan secara teratur pada jarak-jarak tertentu yang telah ditentukan sebelumnya. Dimana pada pengujian ini dilakukan dengan 10 responden berbeda yang dipilih secara acak. Dimana didapatkan hasil seperti berikut. Tujuan dari pengujian ini adalah mengetahui seberapa jauh program yang telah dibuat dapat mendeteksi tangan dengan variasi

jarak yang diujicobakan. Dalam pengujian ini pengukuran dilakukan secara manual dengan kondisi ideal

Tabel 4.2 Pengukuran jarak tangan

Uji ke-	Jarak dalam cm				
	10	30	50	70	90
1	X	V	V	V	X
2	X	V	V	V	X
3	X	X	V	V	V
4	X	X	V	V	V
5	X	V	V	V	X
6	X	V	V	V	V
7	X	V	V	V	V
8	X	V	V	V	X
9	X	X	V	V	X
10	X	V	V	V	X

Dari table Tabel 4.2 dapat didapatkan nilai bervariasi tengtang jarak minimal dan maksimal tangan di depan kamera agar tetap mendapat gesture yang baik. Dimana perbedaan ini terjadi karena tiap responden memeiliki lebar dan banjang tang yang berbeda serta persendingan jari yang berbeda pula. Serta warna kulit tiap responden juga berpengaruh, semakin kontras warna tangan responden dengan background maka semakin mudah pula kamera mengenali gesture pengguna hingga jarak yang relative jauh. Dan dilihat dari data diatas jarak optimum agar gesture tangan terdeteksi adalah pada jarak 50 s/d 70 cm dari kamera.

4.5 Pengujian Akurasi Program

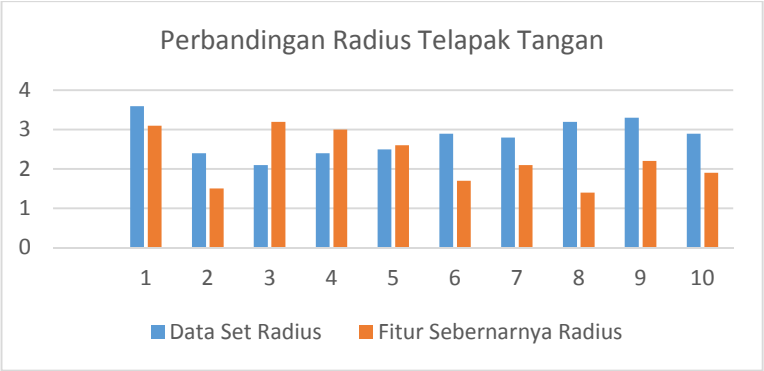
Pengujian akurasi program ini bertujuan untuk menguji keandalan system dalam memprediksi label terhadap bentuk tangan yang dilalukan oleh pengguna.

4.5.1 Pengukuran Parameter Pembelajaran SVM

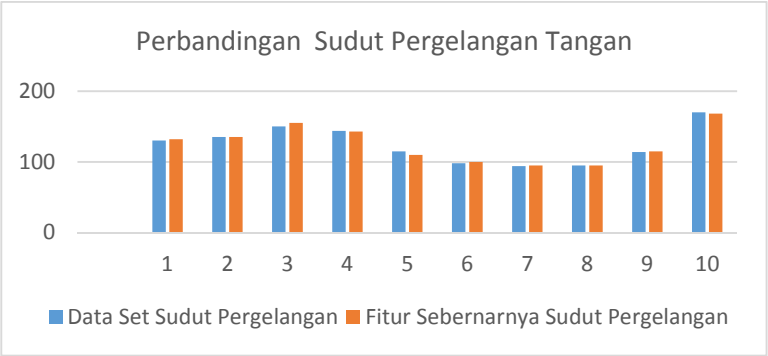
Dengan menggunakan informasi dari layar interface telepon genggam android maka ukuran fitur dari segmentasi citra dapat di estimasi. Ukuran fitur yang dipilih pada pengujian ini ada adalah radius telapak tangan dan sudut pergelangan tangan. Dimana semua gesture yang disegmentasi memilikinya. Pengujian ini dilakukan dengan cara membandingkan fitur yang ada pada data set dan fitur yang didapat langsung oleh pengambilan segmentasi citra pada perubahan gerak tangan.

Tabel 4.3 Perbandingan nilai fitur data set dan data sebenarnya

Nomor yang Uji	Data Set		Fitur Sebernarnya		Nomo yang Terdeteksi
	Radius	Sudut Pergelangan	Radius	Sudut Pergelangan	
1	3.6	130	3.1	132	1
2	2.4	135	1.5	135	2
3	2.1	150	3.2	155	3
4	2.4	144	3	143	4
5	2.5	115	2.6	110	5
6	2.9	98	1.7	100	6
7	2.8	94	2.1	95	7
8	3.2	95	1.4	95	8
9	3.3	114	2.2	115	9
10	2.9	170	1.9	168	10



Gambar 4.3 Grafik perbandingan radius data set dan data sebenarnya.



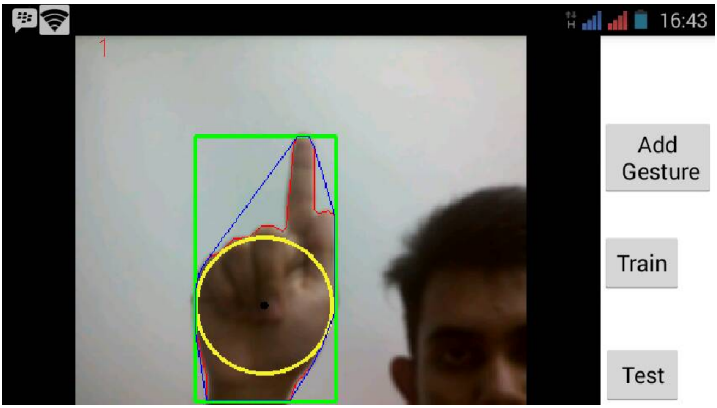
Gambar 4.4 Grafik perbandingan sudut pergelangan tangan data set dan data sebenarnya.

Dilihat pada Tabel 4.3 dapat diketahui perbandingan radius pada data set dan data pengujian sebenarnya sangat berbeda jauh. Ambil saja contoh untuk kasus pendeteksian angka "3". Pada data set radius sebesar 2.1 cm sedangkan pada saat pengujian radius sebesar 3.2 cm. namun angka yang terdeteksi tetaplah angka 3. Dikarenakan meskipun memiliki nilai fitur radius yang berbeda jauh namun kedunya memiliki nilai fitur jumlah jari yang sama serta nilai sudut pergelangan tangan yang tidak terlalu berbeda. Dimana pada data set sudut pergelangan sebesar 150^0 dan

pada pengujian didapatkan sudut pergelangan sebesar 155^0. Sehingga dapat disimpulkan bahwa SVM ini menyesuaikan nilai yang diterima dengan seluruh data set yang ada seperti yang tertera pada perancangan sistem dimana sistem akan mencari data set yang memiliki karakteristik yang mendekati karakteristik fitur hasil pengujian.

4.5.2 Pengujian Akurasi Angka

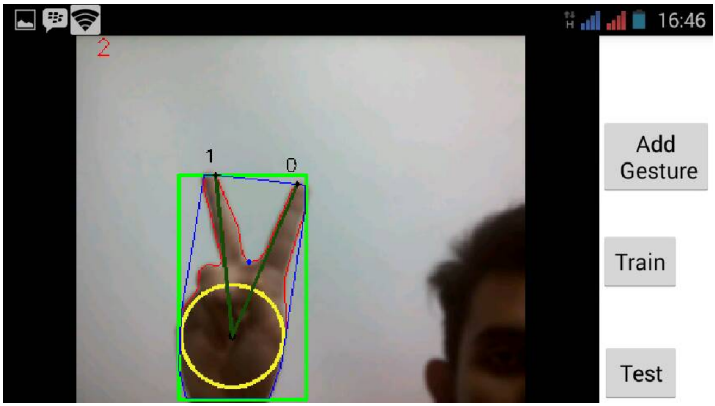
Pada pengujian ini program yang telah dibuat diuji pada waktu kondisi ruangan cukup cahaya, latar belakang homogeny atau satu warna saja dan tidak ada objek lain dibelakangnya. Untuk setiap atau masing-masing angka dilakukan 10 kali percobaan untuk mendapatkan akurasi deteksi baca tangan. Berikut hasil pengujian untuk akurasi angka.



Gambar 4.5 Hasil Pengujian Angka 1

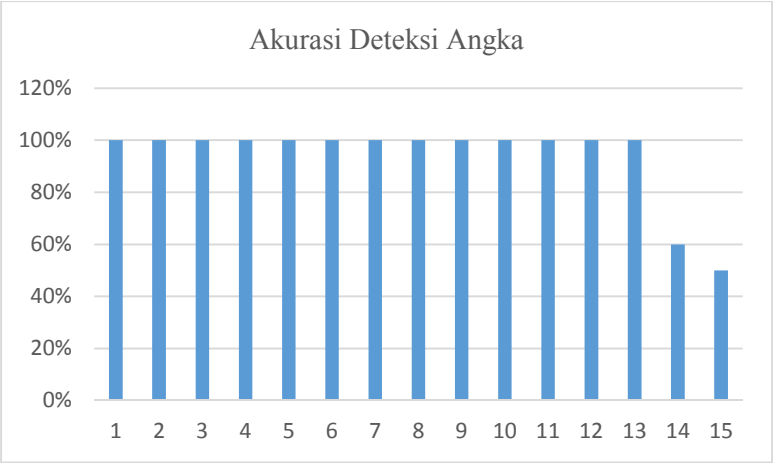
Pada Gambar 4.5 hasil dari pengujian angka satu, fitur yang terekstraksi yaitu radius lingkaran sebesar 2 cm , jari yang terdeteksi 1, sudut pergelangan tangan 135^0 dan jarak kedalaman jari 0.5 cm. sedangkan fitur yang terdapat pada data set yaitu radius lingkaran 3.6 cm, jari yang terdeteksi 1, sudut pergelangan tangan 130^0 dan jarak kedalam jari 0.75 cm. disini dapat disimpulkan bahwa hasil pembelajaran sistem akan mengambil nilai yang mendekati dari fitur yang terdapat pada data set dan juga fitur yang dibelajarkan pada data set adalah bersifat gambar

perspektif. Dimana dicari apakah fitur yang terdeteksi termasuk fungsi pembesaran atau fungsi data set yang diperkecil.



Gambar 4.6 Hasil pengujian angka 2

Begitu halnya pada pengambilan uji sampel untuk angka 2 seperti pada Gambar 4.2. Fitur yang terdeteksi untuk radius lingkaran 1 cm, jumlah jari 2 , sudut pergelangan 140^0 dan kedalam jari 1,5 cm dan 0.4 cm. sedangkan yang ada pada data set yaitu radius lingkaran sebesar 2.4 cm, jumlah jari 2, sudut pergelangan tangan 135 dan kedalaman jari 2,1 cm dan 0,8 cm. dapat disimpulkan dalam pengujian pembelajaran untuk angka 2 sistem mencari fungsi perspektif dari data set input yang diberikan. Dimana nilai yang ada pada data set lebih besar dari data yang dimasukkan sehingga sistem SVM menekstraksi fitur pada masukan dan data set apakah perbandingannya linear atau tidak.



Gambar 4.7 Grafik akurasi deteksi angka

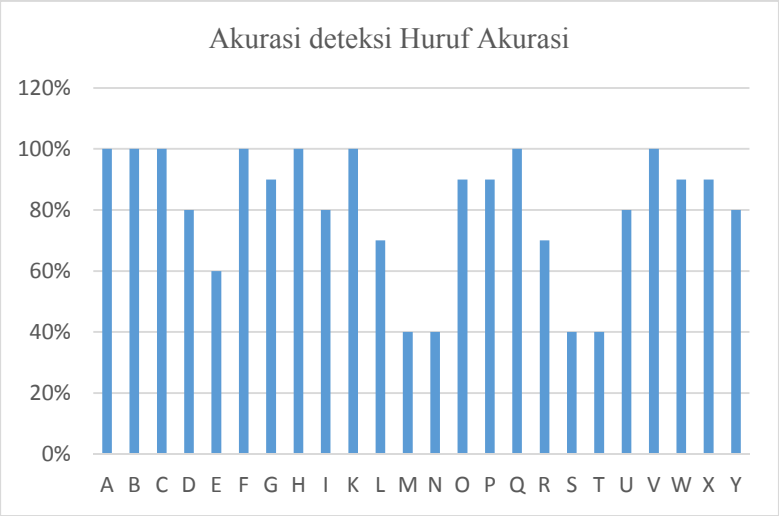
Dari grafik pada Gambar 4.7 diatas didapatkan bahwa program yang telah dibuat memiliki akurasi deteksi baca angka pada tangan sebesar 91.4%. Angka yang memiliki nilai akurasi rendah seperti 14 dan 15 dikarenakan bentuk pola atau postur tangan memiliki kesamaan pola dengan huruf 4 dan 5.

4.5.3 Pengujian Akurasi Huruf

Pada pengujian ini program yang telah dibuat diuji pada waktu kondisi ruangan cukup cahaya, latar belakang homogen atau satu warna saja dan tidak ada objek lain dibelakangnya. Untuk setiap atau masing-masing huruf dilakukan 10 kali percobaan untuk mendapatkan akurasi deteksi baca tangan. Berikut hasil pengujian untuk akurasi deteksi huruf.

Tabel 4.3 Data tes akurasi deteksi huruf

Huruf	Pengujian Ke-									
	1	2	3	4	5	6	7	8	9	10
A	v	v	v	v	v	v	v	v	v	v
B	v	v	v	v	v	v	v	v	v	v
C	v	v	v	v	v	v	v	v	v	v
D	x	v	v	v	v	v	x	v	v	v
E	x	v	v	x	v	v	v	v	x	x
F	v	v	v	v	v	v	v	v	v	v
G	v	x	v	v	v	v	v	v	v	v
H	v	v	v	v	v	v	v	v	v	v
I	v	x	v	v	v	x	v	v	v	v
K	v	v	v	v	v	v	v	v	v	v
L	v	v	v	x	v	v	v	x	x	v
M	x	x	v	x	x	x	v	v	v	x
N	x	x	x	x	v	v	x	x	v	v
O	v	v	v	v	v	v	v	v	x	v
P	v	v	x	v	v	v	v	v	v	v
Q	v	v	v	v	v	v	v	v	v	v
R	v	x	v	v	v	x	v	v	v	x
S	v	v	v	v	v	v	v	v	v	v
T	v	x	x	v	x	v	v	x	x	x
U	v	v	x	x	v	v	v	v	v	v
V	v	v	v	v	v	v	v	v	v	v
W	v	v	v	x	v	v	v	v	v	v
X	v	v	v	v	v	v	v	v	x	v
Y	v	v	v	v	x	v	v	v	v	v



Gambar 4.8 Grafik akurasi deteksi huruf

Dari grafik pada Gambar 4.8 didapatkan bahwa program yang telah dibuat memiliki akurasi deteksi baca tangan sebesar 70,42%. Huruf yang memiliki nilai akurasi rendah seperti huruf M, N, S, T dikarenakan bentuk pola atau postur tangan memiliki kesamaan pola dengan huruf A dan E.

4.6 Pengujian Kata

Pengujian kata ini bertujuan untuk menguji keandalan system dalam memprediksi label terhadap bentuk tangan yang dilalukan oleh pengguna dalam bentuk suatu kata baku yang telah ditentukan sebelumnya. Pada pengujian ini sistem yang telah dibuat diuji pada waktu kondisi ruangan cukup cahaya, latar belakang homogen atau satu warna saja dan tidak ada objek lain dibelakangnya. Untuk setiap atau masing-masing kata dilakukan 5 kali percobaan untuk mendapatkan akurasi deteksi baca tangan. Untuk pengujian kata kata yang diuji yaitu kata “aku” dan “budi” Berikut hasil pengujian untuk akurasi deteksi huruf.

Tabel 4.4 Pengujian kata AKU

Pengujian Ke-	Kata		
	A	K	U
1	v	v	v
2	x	v	x
3	x	x	v
4	v	v	v
5	v	v	v

Tabel 4.5 Pengujian kata DIA

Pengujian Ke-	Kata		
	D	I	A
1	v	v	x
2	v	v	v
3	v	v	v
4	v	v	x
5	v	v	v

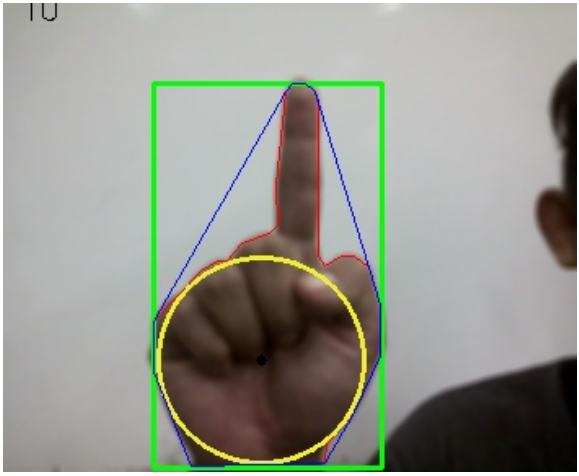
Dari data pada Tabel 4.4 dan Tabel 4.5 diatas, secara keseluruhan system memiliki keandalan 60%. Dimana dari 10x pengujian sistem hanya melakukan 4 kali kesalahan dimana seharusnya U tapi dideteksi sebagai R, A menjadi E. hal ini terjadi karna hasil Contour Convexity huruf R dan huruf U memiliki kesamaan.begitu juga A dan E.

Secara keluruhan hasil Support Vector Machine (SVM) dapat mencari posisi optimal dari hyperplane itu di ruang vektor secara cepat dan tepat. Dan juga SVM dapat mengikuti fungsi dari data training sehingga klasifikasi labeling bisa dilakukan dengan cepat. Berbeda dengan neural network, neural network atau perceptron multiplier membutuhkan waktu lebih panjang untuk menentukan posisi optimal dari hyperplane di ruang vektor karena dilakukan dengan cara mengoreksi nilai weight secara berulang sampai posisi optimal dari hyperplane

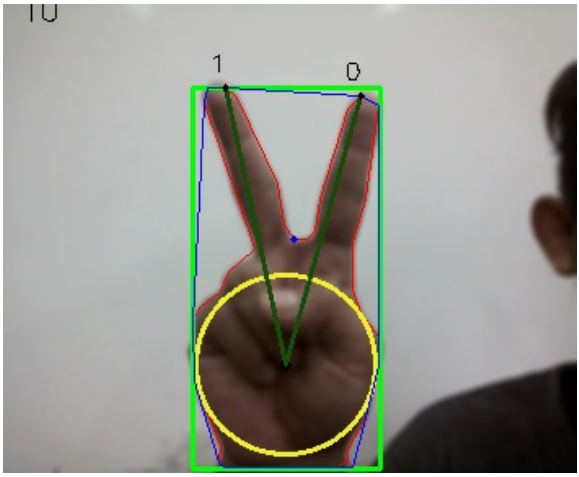
ditemukan hingga dua kelas bisa dipisahkan oleh hyperplane tersebut. Sedangkan pada SVM pemisahanhan dua kelas hanya dilakukan dengan membuat batas vektor saja agar hyperplane optimal bisa ditemukan dan jika kondisi dari dua kelas bercampur pada SVM bisa dilakukan dengan transformasi kernel untuk bisa langsung memisah dua kelas tersebut. Itulah alasan kenapa penulis memilih metode SVM untuk sistem pembelajaran pada Tugas Akhir ini. Karena SVM lebih cepat proses pemisahan dua klas yang berbeda sehingga sistem pembelajaran yang dibuat bisa mendekati real time tracking.

LAMPIRAN

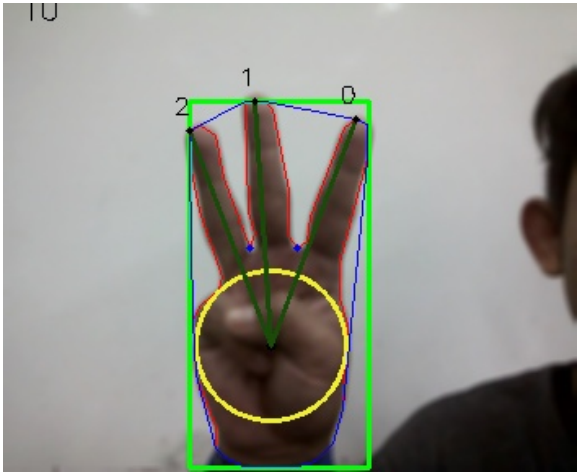
Berikut adalah Data Set yang penulis gunakan:



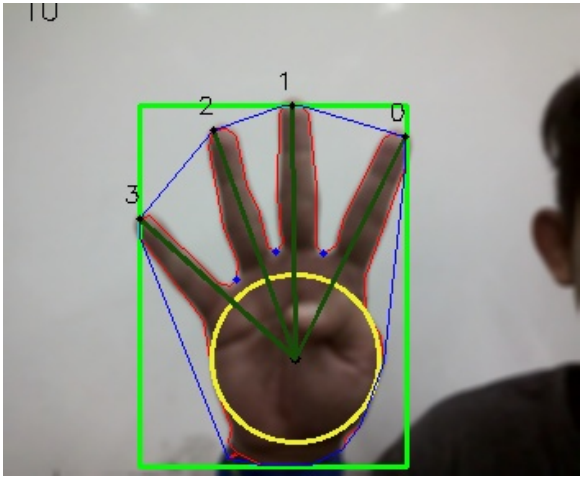
1



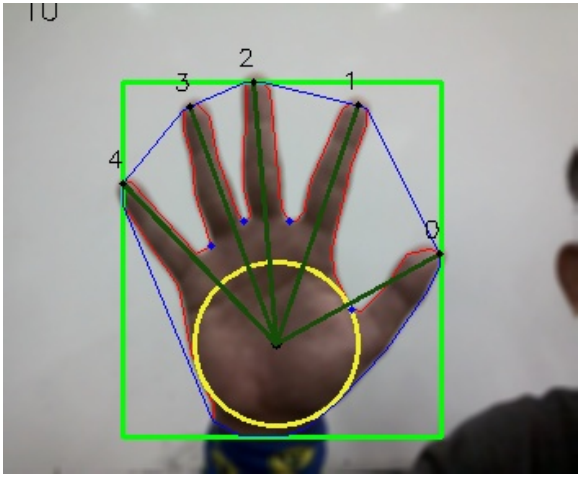
2



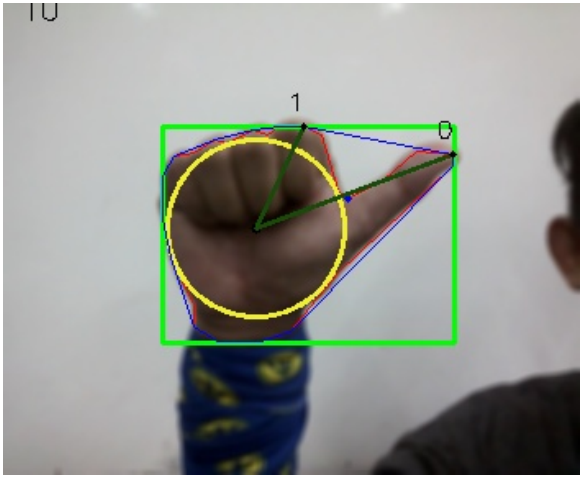
3



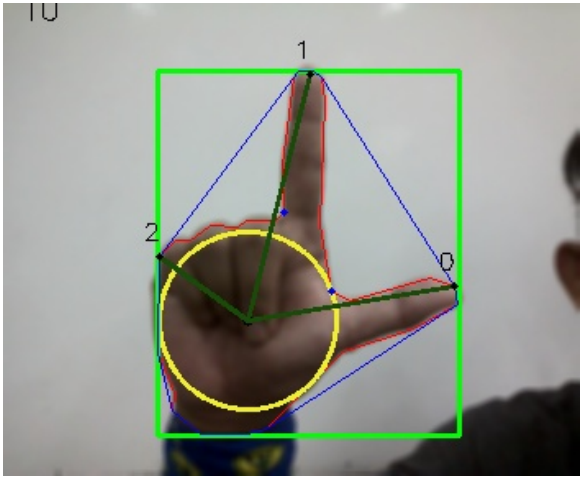
4



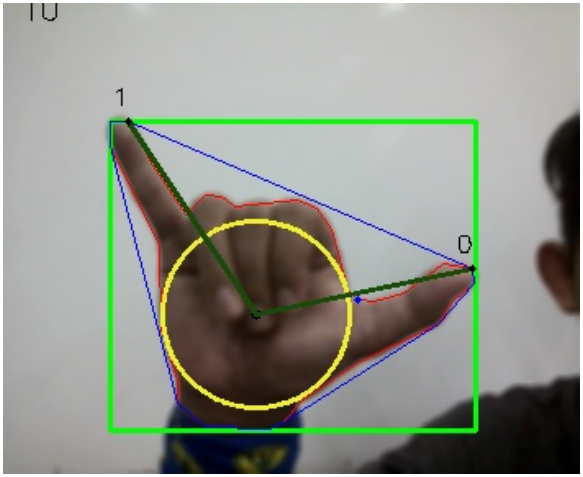
5



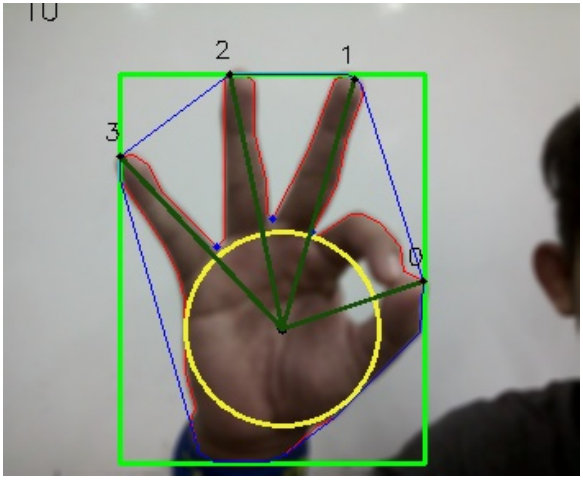
6



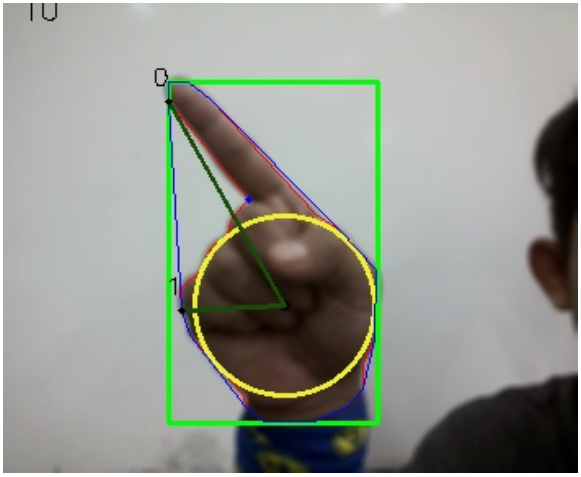
7



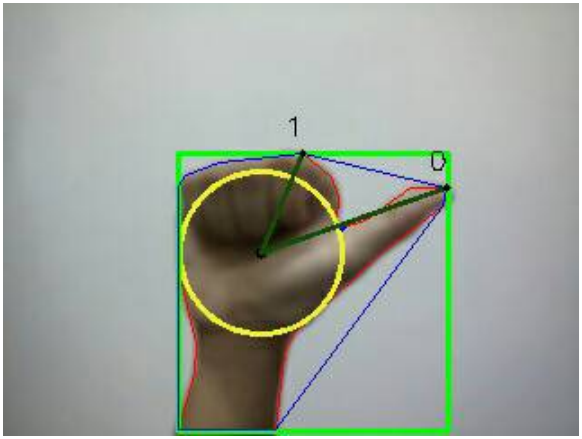
8



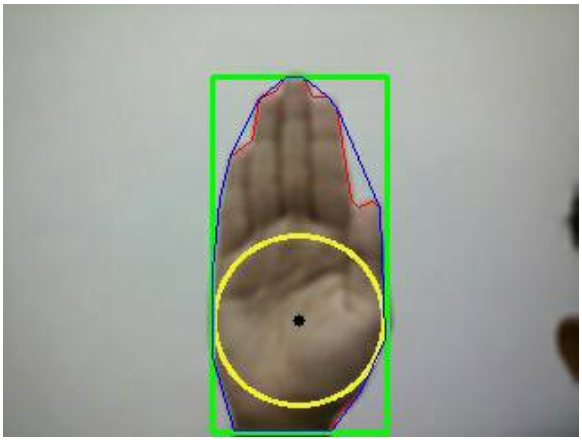
9



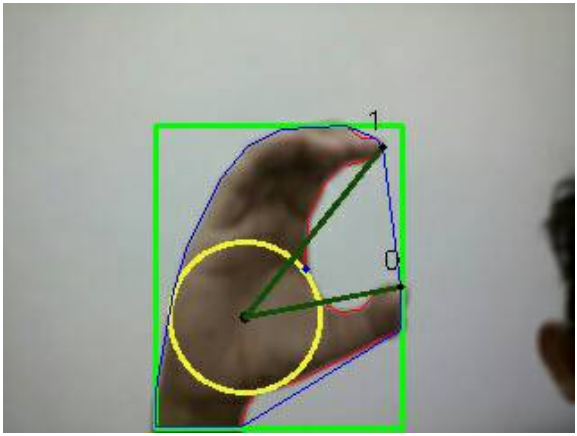
10



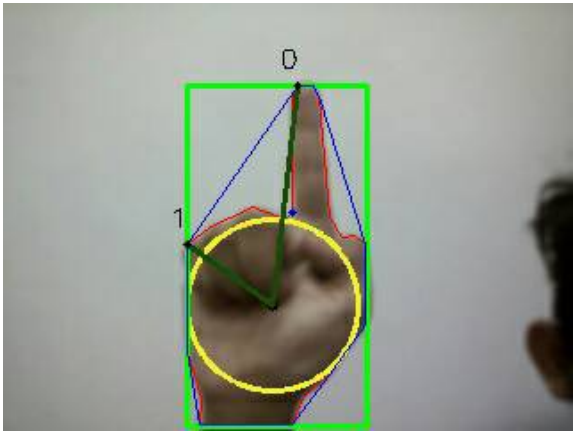
A



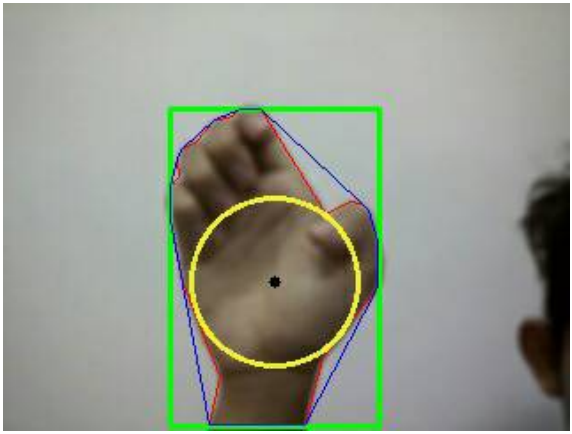
B



C



D



E

BAB V
PENUTUP

Berdasarkan hasil dari uji coba sistem terhadap beberapa subjek, maka dapat ditarik beberapa kesimpulan dan saran untuk pengembangan sistem kedepan.

5.1 Kesimpulan

Kesimpulan yang dapat diambil setelah melakukan uji coba pada sistem yaitu:

1. Dari data rata-rata pengukuran jarak tangan, jarak optimum tangan ke kamera sebesar 50cm. karena pada jarak 50cm dari kamera,kamera bisa mendapatkan fitur tangan secara utuh.
2. Hasil dari pengujian kondisi ideal yaitu pada kondisi pencahayaan cukup dan dengan background yang homogen system memiliki success rate 90%. Dimana system bisa menangkap fitur tangan secara utuh.
3. Pada pencahayaan yang terang error pembacaan fitur tangan mengalami error 25%. Karena cahaya yang terlalu terang menyebabkan efek kekuningan pada tepi tangan yang mengganggu pembacaan fitur tangan.
4. Dari hasil pengujian pada background yang bervariasi system mengalami error 70% karena banyaknya background menyebabkan error pada pengambilan sampling warna.
5. Dari hasil pengujian system lebih bias mengenali fitur tangan untuk angka dari pada fitur tangan untuk huruf. Dikarenakan fitur tangan untuk huruf memiliki fitur yang hamper sama antara fitur tangan satu dan lainnya.
6. Sistem mampu memberikan prediksi label dari perubahan gerak tangan yang didapatkan. Tetapi hasil prediksi label dapat mengalami pergeseran. Pergeseran tersebut disebabkan karena adanya pergeseran nilai estimasi depth dan sudut pergelangan tangan dalam system SVM.

5.2 Saran

Untuk mengembangkan tugas akhir ini penulis memiliki beberapa saran sebagai berikut:

1. Untuk meningkatkan presisi dari estimasi prediksi label, diperlukan penambahan set data uji dengan subjek yang beragam.
2. Metode prediksi Alfabet Sign Languge dengan menggunakan support vector machine dalam diteliti lebih lanjut.
3. Diharapkan kerja sistem tidak terganggu bila ada orang yang berdiri disekitar pengguna.

DAFTAR PUSTAKA

[1] "The Signed Languages of Indonesia: An Enigma," *SIL International*. [Online]. Available: http://www.sil.org/resources/publications/entry/58160. [Accessed: 28-May-2016].

[2] "Android." [Online]. Available: https://www.android.com/intl/id_id/. [Accessed: 08-Jun-2016].

[3] G. R. Bradski and A. Kaehler, *Learning OpenCV: [computer vision with the OpenCV library]*, 1. ed., [Nachdr.]. Beijing: O'Reilly, 2011.

[4] R. M. Gurav and P. K. Kadbe, "Real time finger tracking and contour detection for gesture recognition using OpenCV," 2015, pp. 974–977.

[5] R. Shrivastava, "A hidden Markov model based dynamic hand gesture recognition system using OpenCV," 2013, pp. 947–950.

[6] T. Starner, J. Weaver, and A. Pentland, "Real-time American sign language recognition using desk and wearable computer based video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 12, pp. 1371–1375, Dec. 1998.

[7] B. Demir and S. Ertrk, "Improving SVM classification accuracy using a hierarchical approach for hyperspectral images," 2009, pp. 2849–2852.

[8] Jie Xu, Yuan Yan Tang, Bin Zou, Zongben Xu, Luoqing Li, and Yang Lu, "The Generalization Ability of Online SVM Classification Based on Markov Sampling," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 3, pp. 628–639, Mar. 2015.

[9] A. Mathur and G. M. Foody, "Multiclass and Binary SVM Classification: Implications for Training and Classification Users," *IEEE Geosci. Remote Sens. Lett.*, vol. 5, no. 2, pp. 241–245, Apr. 2008.

[10] B. Demir and S. Ertürk, "Improving SVM classification accuracy using a hierarchical approach for hyperspectral images," in *2009 16th IEEE International Conference on Image Processing (ICIP)*, 2009, pp. 2849–2852.

[11] "Android NDK | Android Developers." [Online]. Available: https://developer.android.com/ndk/index.html?hl=ru. [Accessed: 08-Jun-2016].

[12] “Android Developers.” [Online]. Available: https://developer.android.com/index.html. [Accessed: 08-Jun-2016].
[13] “OpenCV | OpenCV.” .

BIODATA PENULIS

Muhammad Yunus A. lahir di Jombang pada 14 Januari 1994, yang merupakan anak ketiga dari tiga bersaudara dari pasangan Choirul Anam dan Sumiati. Penulis menyelesaikan pendidikan dasar di SDN Mayangan dan dilanjutkan dengan pendidikan menengah di SMPN 1 Jogoroto dan SMAN 2 Jombang. Penulis mengenyam pendidikan perguruan tinggi di jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember (ITS) Surabaya. Selama kuliah, penulis aktif membantu penyelenggaraan kegiatan dan aktif sebagai asisten laboratorium Elektronika Dasar.

Email:
muhammad.yunus.andrian@gmail.com