APLIKASI IDENTIFIKASI BUAH BERDASARKAN WARNA DAN BENTUK MENGGUNAKAN *WEBCAM*

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan Untuk Memenuhi Persyaratan
Memperoleh Gelar Sarjana Teknik

Disusun Oleh:

FANNI KHARISMA

NIM. 0710633064-63

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

UNIVERSITAS BRAWIJAYA

FAKULTAS TEKNIK

MALANG

2012

LEMBAR PERSETUJUAN

APLIKASI IDENTIFIKASI BUAH BERDASARKAN WARNA DAN BENTUK MENGGUNAKAN *WEBCAM*

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:

FANNI KHARISMA

NIM. 0710633064-63

Telah diperiksa dan disetujui oleh :

Dosen Pembimbing I	**Dosen Pembimbing II**
Ir. Muhammad Aswin, MT.	**Adharul Muttaqin, ST., MT.**
NIP. 19640626 199002 1 001	**NIP. 19760121 200501 1 001**

LEMBAR PENGESAHAN

APLIKASI IDENTIFIKASI BUAH BERDASARKAN WARNA DAN BENTUK MENGGUNAKAN *WEBCAM*

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:

FANNI KHARISMA
NIM. 0710633064-63

Skripsi ini telah diuji dan dinyatakan lulus pada
tanggal 24 Januari 2012
DOSEN PENGUJI

Waru Djuriatno, ST., MT.
NIP. 19690725 199702 1 001

R. Arief Setyawan, ST., MT.
NIP. 19760121 200501 1 001

Moch. Rif'an, ST., MT.
NIP. 19710601 200003 1 001

Mengetahui
Ketua Jurusan Teknik Elektro

Dr. Ir. Sholeh Hadi Pramono, MS.
NIP. 19710615 199802 1 003

PENGANTAR

Puji dan syukur penulis panjatkan kepada Allah SWT, karena dengan rahmat, taufik dan hidayah-Nya lah skripsi ini dapat diselesaikan.

Skripsi berjudul "Aplikasi Identifikasi Buah Berdasarkan Warna dan Bentuk Menggunakan *Webcam*" ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Teknik di Jurusan Teknik Elektro Universitas Brawijaya.

Penulis menyadari bahwa penyusunan skripsi ini tidak terlepas dari bantuan berbagai pihak. Oleh karena itu, dengan ketulusan dan kerendahan hati penulis menyampaikan terima kasih kepada:

- Ibunda Nurul Aini, Ayahanda Ali Masdjudi, adikku Dita Fadilla K. atas segala nasehat, kasih sayang, perhatian dan kesabarannya serta telah banyak mendoakan kelancaran penulis hingga terselesaikannya skripsi ini,
- Bapak Dr. Ir. Sholeh Hadi Pramono, MS. selaku Ketua Jurusan dan bapak Muhammad Aziz Muslim, ST., MT., PhD. selaku Sekretaris Jurusan Teknik Elektro Universitas Brawijaya,
- Ibu Rusmi Ambarwati, ST., MT., selaku dosen penasehat akademik yang telah memberikan nasehat, arahan, dan motivasi selama proses akademik penulis,
- Bapak Waru Djuriatno, ST., MT., selaku Ketua Kelompok Dosen Keahlian Rekayasa Komputer Jurusan Teknik Elektro Universitas Brawijaya,
- Bapak Ir. Muhammad Aswin., MT., dan Bapak Adharul Muttaqin, ST., MT., selaku Dosen Pembimbing atas segala bimbingan, nasehat, pengarahan, motivasi, saran dan masukan yang telah diberikan dalam pengerjaan skripsi,
- Upik, Adit, Iwenk, Benny, Ronny, Adi, Ana, yang merupakan sahabat kecil penulis selama ini,
- Aulia Wildan, ST., Wirawan Hidayat, Boby Septian S., Ichwan Maulana, Wildan Khoirurrizqi, Gallant Hendrayana, Dwyan Zakaria, dan Ida Bagus Willy, yang selalu memberi doa serta menemani saya dalam suka dan duka selama menjalani masa perkuliahan di Teknik Elektro Universitas Brawijaya,
- Teman-teman Laboratorium Dasar Elektrik dan Pengukuran, Core-nerz '07, teman-teman di kelembagaan, senior serta semua pihak yang tidak mungkin bagi penulis untuk mencantumkan namanya satu-persatu, terima kasih banyak atas bantuan dan dukungannya.

Pada akhirnya, penulis menyadari bahwa skripsi ini masih belum sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun. Penulis berharap semoga skripsi ini dapat bermanfaat bagi pengembangan ilmu pengetahuan dan teknologi.

Malang, November 2011

Penulis

ABSTRAK

Fanni Kharisma, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, Aplikasi Identifikasi Buah Berdasarkan Warna dan Bentuk Menggunakan *Webcam*, Dosen Pembimbing: Ir. Muhammad Aswin, MT., dan Adharul Muttaqin, ST., MT.

Membedakan suatu objek dapat dilakukan dengan mengimplementasikan metode pengenalan pola yang merupakan bagian dari ilmu komputer. Pengenalan pola merupakan proses mengkategorikan setiap sampel data yang telah diukur atau diamati sebagai anggota salah satu dari beberapa kelas atau kategori. Salah satu metodenya adalah jaringan saraf tiruan. Setiap jenis buah-buahan memiliki karakteristik yang membedakan antara buah yang satu dengan buah yang lain. Untuk itu dapat dibuat sebuah sistem komputer yang dapat mengidentifikasi jenis buah-buahan dengan menerapkan pengenalan pola tersebut.

Skripsi ini menguraikan pembuatan suatu aplikasi untuk mengidentifikasi jenis buah-buahan berdasarkan warna dan bentuk menggunakan webcam. Citra ditangkap oleh *webcam* kemudian dilakukan proses ekstraksi ciri. Ciri yang diambil adalah rata-rata nilai RGB dan *roundness*. Selanjutnya dilakukan pelatihan jaringan saraf tiruan menggunakan metode LVQ dengan tujuan memperoleh nilai bobot yang mewakili setiap jenis buah serta mengetahui parameter maksimal epoh, *learning rate*, dan pengurangan *learning rate* yang ideal. Proses identifikasi pada aplikasi ini menghasilkan presentase keberhasilan pengenalan sebesar 73.9% dari 23 buah yang diuji dengan parameter maksimal epoh=50, *learning rate*=0.9, dan pengurangan *learning rate*=0.01.

Kata kunci: buah, jaringan saraf tiruan, ekstraksi ciri, LVQ

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Buah merupakan salah satu sumber daya alam yang memberikan banyak manfaat bagi kehidupan manusia terutama di Indonesia yang letak geografisnya menunjang untuk meningkatkan taraf hidup masyarakatnya. Buah memiliki berbagai jenis dengan karakteristik yang membedakannya. Dengan semakin pesatnya perkembangan teknologi membuat jenis buah semakin banyak dikembangkan oleh para ahli. Hal ini tentunya semakin menyulitkan manusia dalam membedakan satu jenis buah dengan buah yang lain.

Komputer merupakan alat yang dipakai untuk mengolah data menurut prosedur yang dirumuskan. Komputer banyak digunakan untuk mempermudah suatu pekerjaan manusia karena dapat bekerja dengan cepat. Saat ini perkembangan teknologi komputer yang pesat serta kebutuhan akan komputer itu sendiri semakin tinggi. Salah satu bidang ilmu komputer yang mengalami perkembangan dan banyak digunakan adalah pengenalan pola.

Pengenalan pola merupakan suatu ilmu komputer untuk mengklasifikasikan atau menggambarkan suatu objek. Teknologi pengenalan pola yang dapat melakukan klasifikasi atau penggambaran suatu objek yang memiliki kemampuan seperti otak manusia adalah jaringan saraf tiruan. Jaringan saraf tiruan merupakan salah satu teknologi pengenalan pola yang meniru prinsip kerja saraf manusia sehingga mampu untuk mempelajari dan menyelesaikan suatu permasalahan sesuai dengan yang diinginkan.

Dengan menerapkan teknologi ini tentunya diharapkan dapat membantu manusia untuk mengenali suatu jenis buah dalam berbagai bidang seperti bidang perdagangan, pendidikan maupun pertanian.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan diatas, maka rumusan masalah ditekankan pada:

1. Bagaimana melakukan ektraksi ciri pada citra buah
2. Bagaimana mengimplementasikan algoritma LVQ

1.3 Batasan Masalah

Beberapa hal yang menjadi batasan dalam pembuatan program ini antara lain:

1. Data yang ditangkap oleh *webcam* berupa citra buah.
2. Proses pengambilan gambar dilakukan dengan posisi kamera webcam berada tepat di depan objek dengan jarak 20 cm dan tidak terhalangi objek lain dengan *background* objek warna putih serta pada penerangan yang cukup.
3. Pengambilan gambar buah merupakan bagian membujur dari objek saat luas proyeksi terbesar.
4. Jumlah buah untuk pelatihan adalah 100 buah dengan variasi 5 buah untuk masing-masing buah dan untuk pengujian adalah 23 buah.
5. Buah yang digunakan adalah buah matang yang utuh secara fisik dengan jenis apel manalagi, mangga manalagi, pisang susu, jambu merah, dan jeruk manis.
6. Kamera yang digunakan beresolusi 320x240 pixel dengan format penyimpanan file adalah bitmap (.bmp).
7. Metode pengambilan keputusan menggunakan jaringan syarat tiruan *Learning Vector Quantization*.

1.4 Tujuan

Tujuan penyusunan skripsi ini adalah:

Merancang dan membangun suatu aplikasi yang dapat mengidentifikasi buah dengan webcam.

1.5 Sistematika Penulisan

Penulisan skripsi dibagi dalam tujuh bab dengan sistematika penulisan sebagai berikut :

BAB I PENDAHULUAN

Memuat latar belakang, rumusan masalah, batasan masalah, tujuan penulisan, manfaat dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Membahas kajian pustaka dan dasar teori yang digunakan pada skripsi ini.

BAB III METODE PENELITIAN

Membahas metode yang digunakan dalam skripsi ini serta langkah-langkah yang diambil.

BAB IV PERANCANGAN DAN IMPLEMENTASI

Menjelaskan langkah-langkah perancangan aplikasi identifikasi buah beserta penjelasan algoritma yang digunakan.

BAB V PENGUJIAN DAN ANALISIS

Pengujian perangkat lunak dan analisis hasil pengujian.

BAB VI PENUTUP

Memuat kesimpulan yang diperoleh serta saran untuk pengembangan lebih lanjut.

BAB II
TINJAUAN PUSTAKA

2.1 Buah

Dalam pandangan botani, buah adalah organ pada tumbuhan berbunga yang merupakan perkembangan lanjutan dari bakal buah (ovarium). Buah biasanya membungkus dan melindungi biji. Aneka rupa dan bentuk buah tidak terlepas kaitannya dengan fungsi utama buah, yakni sebagai pemencar biji tumbuhan.

Pada banyak spesies tumbuhan, yang disebut buah mencakup bakal buah yang telah berkembang lanjut beserta dengan jaringan yang mengelilinginya. Bagi tumbuhan berbunga, buah adalah alat untuk menyebar luaskan biji-bijinya. Adanya biji di dalam dapat mengindikasikan bahwa organ tersebut adalah buah, meski ada pula biji yang tidak berasal dari buah.

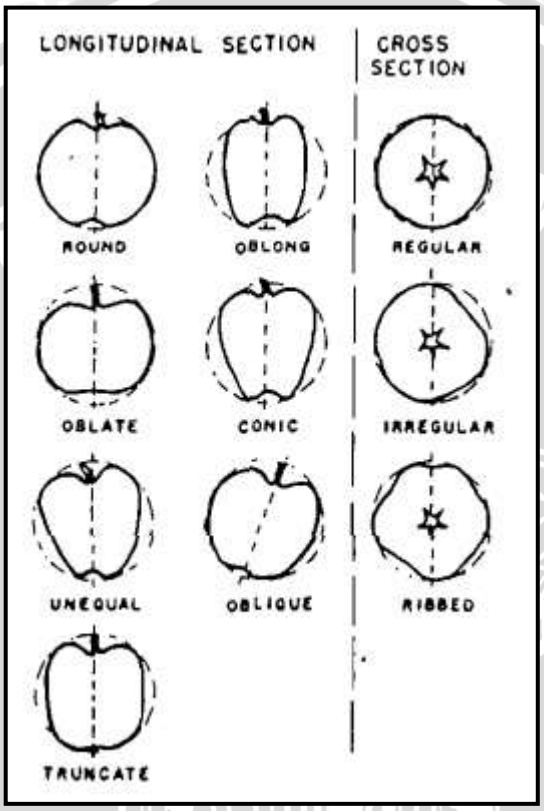
Buah dalam pengertian hortikultura atau pangan merupakan pengertian yang dipakai oleh masyarakat luas. Dalam pengertian ini, batasan buah menjadi longgar. Istilah "buah-buahan" dapat digunakan untuk pengertian demikian. Buah-buahan adalah setiap bagian tumbuhan di permukaan tanah yang tumbuh membesar dan (biasanya) berdaging atau banyak mengandung air. Buah dalam pengertian ini tidak terbatas yang terbentuk dari bakal buah, melainkan dapat pula berasal dari perkembangan organ yang lain. Karena itu, untuk membedakannya, buah yang sesuai menurut pengertian botani biasa disebut buah sejati.

Buah merupakan salah satu hasil pertanian yang tentunya memiliki sifat-sifat bahan pertanian. Sifat-sifat bahan pertanian antara lain sifat fisik, mekanis, panas, listrik, dan optik. Karakteristik fisik bahan pertanian berkaitan dengan bentuk dan ukuran.

Bentuk dari biji-bijian, buah-buahan dan tanaman pada umumnya tidak beraturan (irregular), sedemikian sehingga jumlah data pengukuran yang sangat besar diperlukan untuk mendiskripsikannya secara akurat. Meski demikian, dari praktek pengukuran-pengukuran menunjukkan bahwa keragaman bentuk secara umum dapat dijelaskan dengan baik oleh poros orthogonal yang ditentukan sesuai tujuan misalnya, benih biasanya dikarakteristikkan oleh panjang, lebar dan tebal. Pada beberapa kasus, karakteristik suatu produk cukup dinyatakan dengan dua atau bahkan satu dimensi saja.

Ukuran bahan-bahan pertanian tidak seragam, melainkan tersebar disekitar nilai reratanya. Dengan demikian, sangatlah penting untuk menyatakan distribusi dari suatu ukuran disamping reratanya.

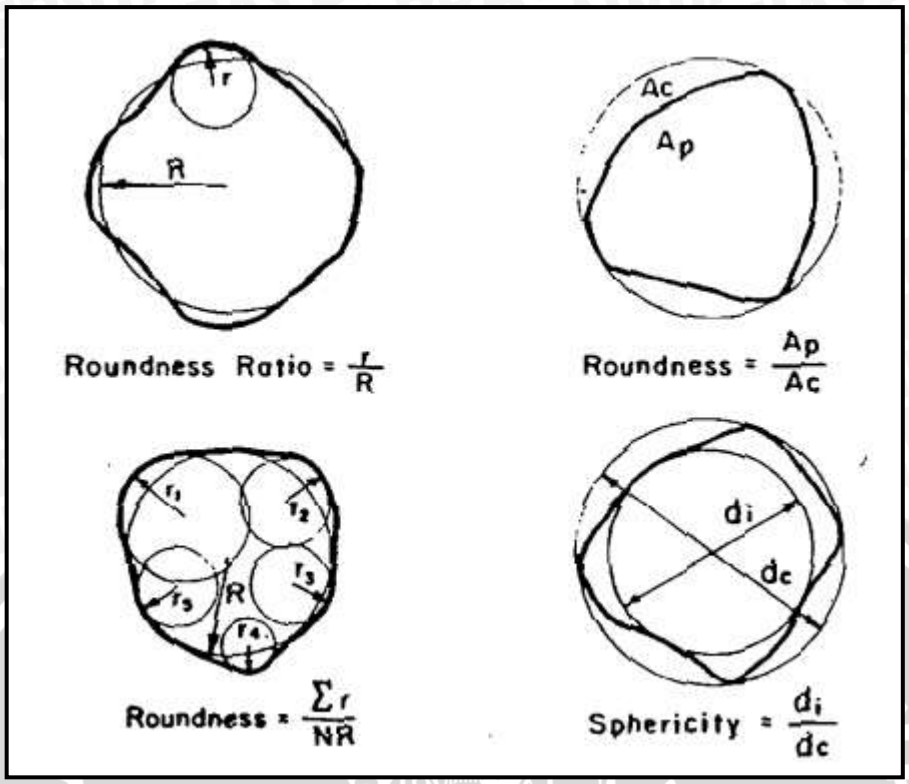
Secara umum, belum ditemukan suatu metode terapan untuk menyatakan dengan tepat bentuk dari produk-produk pertanian. Bentuk suatu tanaman dan produknya pada umumnya diperbandingkan dengan basis dimensi longitudinal dan penampang melintang, untuk memetakan bentuk-bentuk standarnya.



Gambar 2. 1 Contoh pendeskripsian bentuk buah

(**Sumber:** Nuri N. Mohsenin, 1986)

Komparasi visual bentuk suatu produk terhadap bentuk standar adalah sangat mudah tetapi tidak lepas dari kesalahan, tergantung dari subyektivitas pengamat. Karenanya, dalam kasus dimana teknologi proses sangat dipengaruhi oleh bentuk, penggunaan indek pengukuran yang obyektif sangat disarankan. Tujuan dari pengukuran indeks untuk membuat karakteristik suatu bentuk mungkin berbeda, tergantung bahan dan tujuan perlakuan. Indeks-indeks yang umum digunakan adalah kebundaran (roundness), rasio kebundaran (roundness ratio), kebulatan (sphericity), rasio aksial (axial ratio), derajat ketidaksamaan (degree of inequality), dll.



Gambar 2. 2 Contoh indeks pengukuran

(**Sumber:** Nuri N. Mohsenin, 1986)

Sifat fisik bahan pertanian tentunya terdapat pada berbagai buah, seperti apel manalagi, jambu merah, mangga manalagi, jeruk manis dan pisang susu. Apel manalagi memiliki tekstur agak liat dan kurang kandungan airnya. Warna daging buahnya putih kekuningan. Buahnya berbentuk agak bulat dengan ujung dan pangkal berlekuk dangkal. Diameter buah antara 4-7 cm dan berat 75-160 g per buah. Kulit buahnya berwarna hijau muda kekuningan saat matang.

Jeruk manis buahnya berbentuk bulat pendek atau agak bulat dengan ukuran rata-rata 5,7 x 6,3 cm. Kulit buah matang berwarna kuning dan permukaannya halus. Ujung buah berlekuk dalam. Buah jeruk ini tidak berpusar buah. Ketebalan kulitnya sekitar 2,3 mm. Daging buah bertekstur lunak dengan rasa manis. Buahnya mengandung banyak air. Berat buah rata-rata 123,3 g per buah.

Mangga manalagi rasa buahnya manis dan aromanya harum. Bentuk buahnya jorong, pangkal meruncing, ujung membulat, dan berparuh jelas. Warna buah kuning pada pangkalnya dan hijau pada ujungnya. Kulit buah tebal dengan lapisan lilin dengan bintik-bintik kelenjar yang keputihan. Daging buah tebal, berwarna kuning menarik, teksturnya lembut dan lunak, dan tidak banyak mengandung air. Rasanya manis segar,

seperti perpaduan antara rasa mangga arumanis dan golek. Panjang buahnya sekitar 14 cm dan bobotnya lebih dari 500 g.

Pisang susu memiliki bentuk buah bulat memanjang dengan panjang rata-rata sekitar 11,5 cm dan berat berkisar 50 g. Daging buahnya berwarna putih kekuningan dan rasanya manis. Kulit buahnya tipis berwarna kuning cerah. Buah jambu merah memiliki warna kulit hijau kekuningan, warna daging buah merah, rasa buahnya manis dan mengandung air. Buah berdompolan, bentuknya bulat telur, lonjong atau berbentuk buah pir, dengan ukuran beragam diameter sekitar 2,5-10.

2.2 Citra Digital

Citra digital adalah citra yang bisa diolah langsung oleh komputer dan tersimpan dalam media simpan digital misalnya memori, harddisk, CD, dll. Citra digital merupakan sebuah larik (*array*) yang berisi nilai-nilai real maupun komplek yang direpresentasikan dengan bit tertentu. Citra digital didefinisikan sebagai fungsi *f(x,y) berukuran M baris dan N kolom,* dimana *x* dan *y* adalah koordinat spasial dan nilai *f(x,y)* adalah intensitas citra pada koordinat tersebut. Citra digital dapat ditulis dalam bentuk matrik sebagai berikut:

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \quad (2\text{-}1)$$

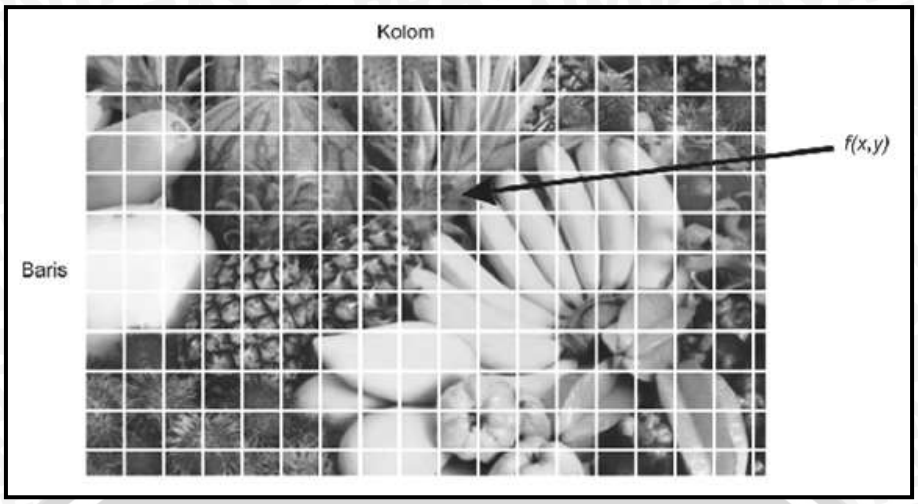
dengan:

f(x, y) = intensitas pixel pada posisi x dan y

M = jumlah kolom

N = jumlah baris

Nilai pada suatu irisan antara baris dan kolom (pada posisi x,y) disebut dengan pixel.



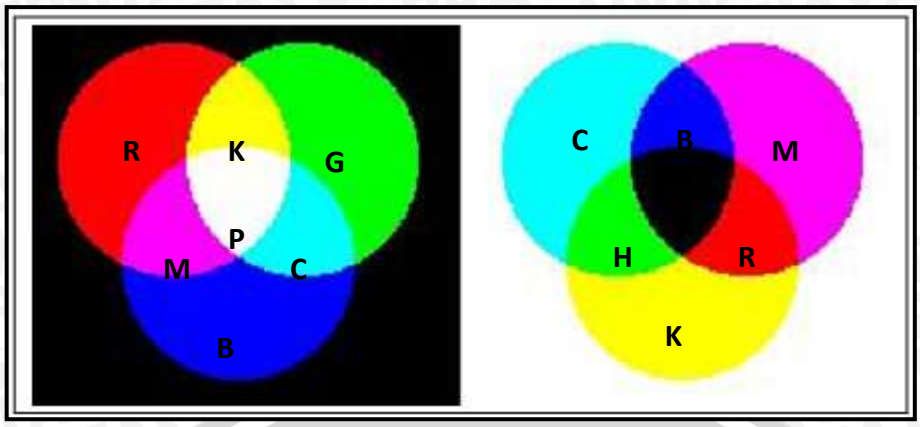
Gambar 2. 3 Ilustrasi digitalisasi citra (pixel posisi x=10, y=4 memiliki nilai 108)

(**Sumber:** Darma Putra, 2010)

2.3 Citra Berwarna

Citra berwarna merupakan gabungan dari beberapa lapis citra kanal warna. Teknologi dasar untuk menciptakan warna pada citra digital merupakan kombinasi dari kanal tiga warna dasar yaitu merah, hijau, dan biru (*RGB- Red, Green, Blue*) ditunjukkan pada Gambar 2.4. Kombinasi kanal warna ini didapat dari penangkapan intensitas cahaya pada frekuensi yang berbeda, yaitu frekuensi 560 nm (merah), 530 nm (hijau) dan 430 nm (biru). Pencampuran warna akan menghasilkan warna baru, seperti pada gambar 2.4.

Pada latar belakang warna hitam, warna merah dicampur dengan warna hijau menghasilkan warna kuning, warna merah dicampur dengan warna biru menghasilkan warna magenta, warna hijau dicampur dengan warna biru menghasilkan warna cyan, dan ketika warna merah, hijau dan biru dicampur akan menjadi warna putih. Sedangkan pada latar belakang warna putih, warna cyan dicampur dengan warna magenta menghasilkan warna biru, warna cyan dicampur dengan warna kuning menghasilkan warna hijau, warna magenta dicampur dengan warna kuning menghasilkan warna merah, dan ketika warna cyan, magenta, dan kuning dicampur akan menjadi warna hitam.



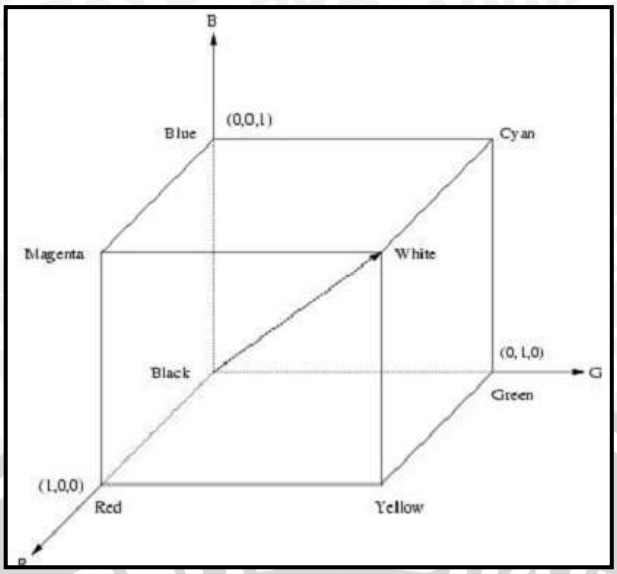
Gambar 2. 4 Penambahan campuran warna merah, hijau dan biru

(**Sumber:** Adang Suhendra)

Prinsip dasar pengolahan citra adalah pengolahan warna RGB pada posisi tertentu. Dalam pengolahan citra, warna direpresentasikan dengan nilai hexadesimal dari 0x00000000 sampai 0x00ffffff. Definisi nilai warna tersebut berisikan variabel 0x00 yang menyatakan angka dibelakangnya adalah hexadesimal. Setiap warna mempunyai range nilai 00 (angka desimal bernilai 0) sampai ff (angka desimal bernilai 255) atau mempunyai derajat keabu-abuan 256 = 2^8, dengan demikian range warna RGB adalah (2^8) (2^8) (2^8) = 2^{24} atau dikenal dalam windows dengan istilah *True Colour*.

2.4 Model Warna RGB

Suatu citra dalam model RGB terdiri dari tiga bidang citra yang saling lepas, masing-masing terdiri dari warna utama: merah, hijau dan biru. Suatu warna dispesifikasikan sebagai campuran sejumlah komponen warna utama. Gambar 2.5 menunjukkan bentuk geometri dari model warna RGB untuk menspesifikasikan warna menggunakan sistem koordinat *cartesian.* Spektrum *grayscale* (tingkat keabuan) yaitu warna yang dibentuk dari gabungan tiga warna utama dengan jumlah yang sama, berada pada garis yang menghubungkan titik hitam dan putih.



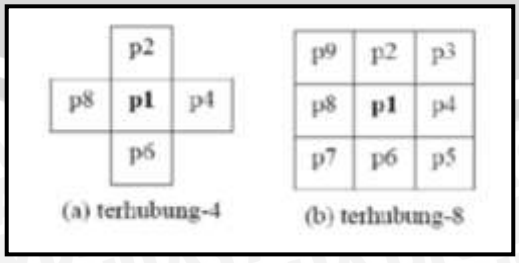
Gambar 2. 5 Koordinat warna RGB

(**Sumber:** Rafael C. Gonzales and Richards E. Woods, 2010)

2.5 Operasi Morfologi Citra

Operasi morfologi adalah teknik pengolahan citra yang didasarkan pada bentuk segmen atau region dalam citra. Karena difokuskan pada bentuk obyek, maka operasi inibiasanya diterapkan pada citra biner. Meskipun lebih banyak dipakai pada citra biner, operasi morfologi sering pula digunakan pada citra skala keabuan dan warna. Secara umum, pemrosesan citra secara morfologi dilakukan dengan cara mempassing sebuah *structuring element* terhadap sebuah citra dengan cara yang hampir sama dengan konvolusi. *Structuring element (SE)* dapat diibaratkan dengan mask pada pemrosesan citra biasa (bukan secara morfologi).

SE merupakan suatu matrik dan pada umumnya berukuran lebih kecil. Pada gambar 2.6 menyatakan contoh SE untuk operasi terhubung-4 (*4-connected*) tetangga dan operasi terhubung-8 (*8-connected*) tetangga.



Gambar 2. 6 Beberapa contoh SE

(**Sumber:** Emy, 2007)

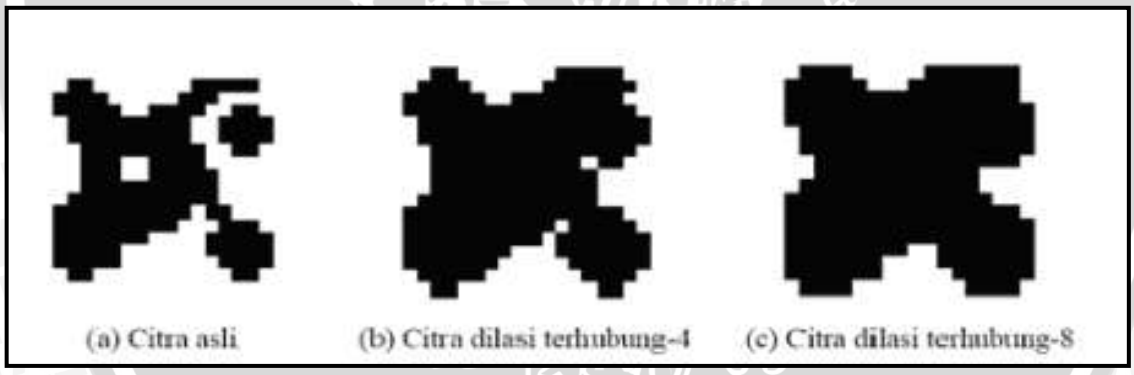
Ada 2 operasi dasar morfologi yaitu dilasi dan erosi. Kedua operasi dasar tersebut menjadi basis untuk membuat berbagai operasi morfologi yang sangat berguna untuk pengolahan citra digital, seperti *opening*, *closing*, *hit and miss transform*, *thinning*, *thickening*, dan *boundary extraction*.

2.5.1 Dilasi

Operasi dilasi dilakukan untuk memperbesar ukuran segmen obyek dengan menambah lapisan di sekeliling obyek. Bila suatu objek (citra input) dinyatakan dengan A dan SE dinyatakan dengan B serta Bx menyatakan translasi B sedemikian sehingga pusat B terletak pada x. Operasi dilasi A dengan B dapat dinyatakan sebagai berikut.

$$D(A,B) = A \oplus B = \{x : B_x A \neq \emptyset\} \quad (2\text{-}2)$$

Dengan ø menyatakan himpunan kosong. Terdapat 2 cara untuk melakukan operasi ini, yaitu dengan cara mengubah semua titik latar yang bertetangga dengan titik batas menjadi titik obyek, atau lebih mudahnya set setiap titik yang tetangganya adalah titik obyek menjadi titik obyek dan dengan mengubah semua titik di sekeliling titik batas menjadi titik obyek, atau lebih mudahnya set semua titik tetangga sebuah titik obyek menjadi titik obyek. Gambar 2.7 merupakan contoh dari operasi dilasi.



Gambar 2. 7 Proses dilasi citra

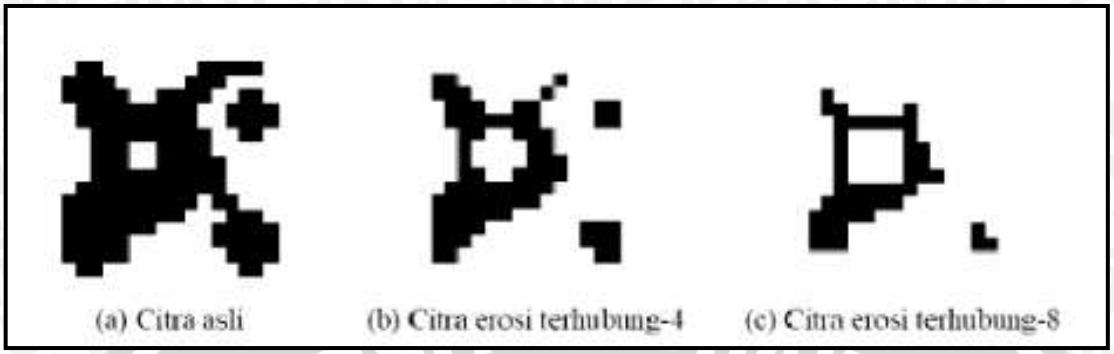
(**Sumber:** Emy, 2007)

2.5.2 Erosi

Erosi merupakan proses penghapusan titik-titik batas objek menjadi bagian dari latar, berdasarkan structuring element yang digunakan. Operasi erosi dapat dinyatakan sebagai berikut.

$$D(A,B) = A \ominus B = \{x : B_x \subset X\} \quad (2\text{-}3)$$

Pada operasi ini, ukuran obyek diperkecil dengan mengikis sekeliling objek. Cara yang dapat dilakukan ada 2 yaitu, dengan mengubah semua titik batas menjadi titik latar dan dengan menset semua titik di sekeliling titik latar menjadi titik latar. Gambar 2.8 merupakan contoh erosi citra.



Gambar 2. 8 Proses erosi citra

(**Sumber:** Emy, 2007)

2.5.3 Opening

Operasi pembukaan merupakan kombinasi antara operasi erosi dan dilasi yang dilakukan secara berurutan, tetapi citra asli dierosi terlebih dahulu baru kemudian hasilnya didilasi. Operasi opening cenderung akan memperhalus objek pada citra, memutus sambungan yang sempit (break narrow joins), dan menghilangkan efek pelebaran pada objek (remove protrusions). Gambar 2.9 merupakan contoh dari opening citra.

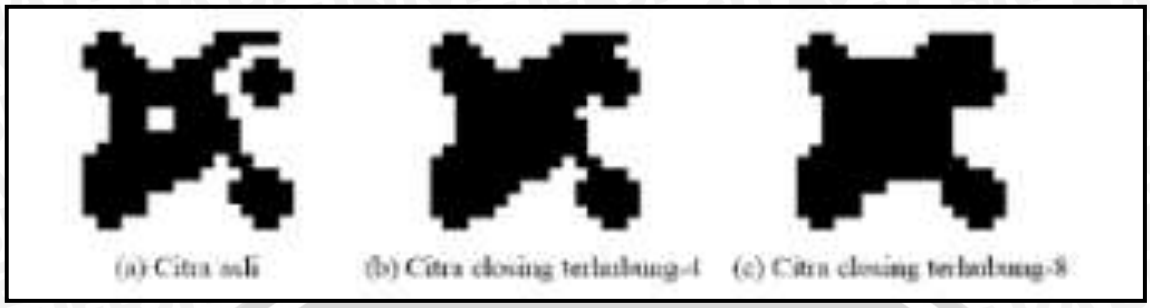
Gambar 2. 9 Proses opening citra

(**Sumber:** Emy, 2007)

2.5.4 Closing

Closing adalah proses dilasi yang diikuti dengan erosi. Citra asli didilasi terlebih dahulu, kemudian hasilnya dierosi. Operasi closing juga cenderung akan

memperhalus objek pada citra, namun dengan cara menyambung pecahan-pecahan (fuses narrow breaks and thin gulf) dan menghilangkan lubang-lubang kecil pada objek. Gambar 2.10 merupakan contoh dari operasi closing citra.

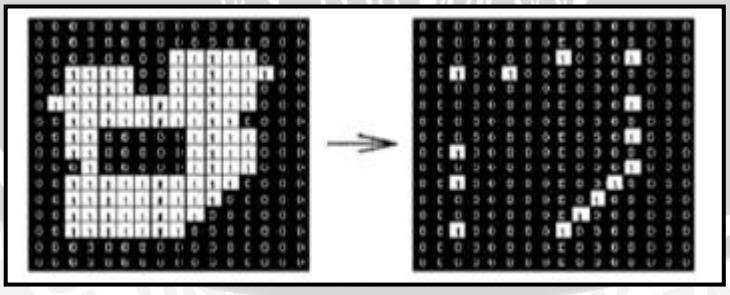


Gambar 2. 10 Proses closing citra

(**Sumber:** Emy, *2007*)

2.5.5 Hit and miss transform

Transformasi *hit-and-miss* merupakan metode dasar morfoloagi untuk mendeteksi bentuk. Proses hit-and-miss dapat dilakukan dengan membandingkan setiap pixel citra input dengan nilai pusat SE dengan cara melapiskan SE dengan citra sehingga pusat SE tepat dengan posisi pixel citra yang diproses. Jika semua pixel pada SE tepat sama dengan semua nilai pixel citra maka pixel input diset nilainya dengan nilai *foreground*, bila tidak maka input pixel diberi nilai pixel *background*. Setelah mendapatkan citra hasil hit and miss untuk masing-masing kernel, kemudian dilanjutkan dengan operasi OR pada semua citra secara bersamaan untuk mendapatkan hasil akhir yang menunjukkan semua orientasi sudut. Gambar 2.11 menunjukkan contoh hit and miss.



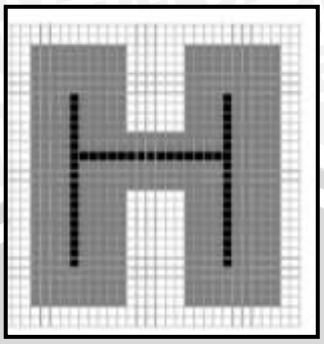
Gambar 2. 11 Proses hit-and-miss transform

(**Sumber:** Darma Putra, 2010)

2.5.6 Thinning

Thinning merupakan suatu operasi morfologi yang mengubah bentuk asli citra biner menjadi citra yang menampilan batas-batas objek hanya setebal 1 pixel. Tujuan

thinning adalah untuk menghilangkan pixel tertentu pada objek citra biner. thinning tidak boleh menghilangkan *end-point*, memutus koneksi yang ada, mengakibatkan *excessive* erosi. Gambar 2.12 menunjukkan contoh dari operasi thinning.

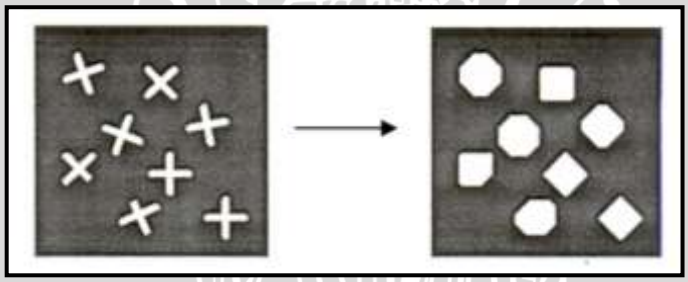


Gambar 2. 12 Operasi thinning

(**Sumber:** Emy, 2007)

2.5.7 Thickening

Thickening digunakan untuk memperluas daerah dari suatu objek. Operasi ini dilakukan untuk menambahkan beberapa wilayah objek dalam citra biner. Operasi thickening sering digunakan untuk menentukan aproksimasi convex hull dari suatu objek. Gambar 2.13 menyatakan contoh dari proses thickening.



Gambar 2. 13 Operasi thickening

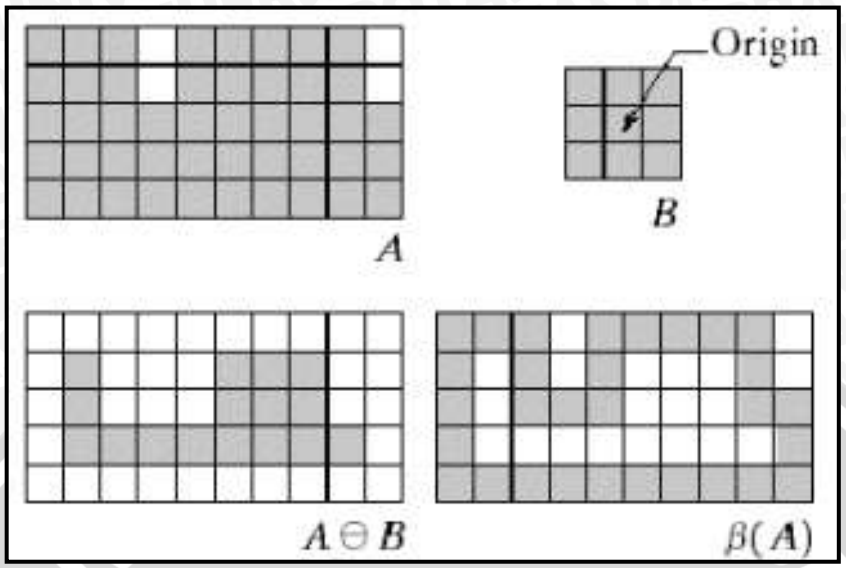
(**Sumber:** Darma Putra, 2010)

2.5.8 Boundary detection

Salah satu aplikasi operasi erosi dan dilasi adalah untuk deteksi bidang batas suatu objek dalam citra. Jika A adalah suatu citra dan B adalah suatu structuring element yang kecil yang terdiri atas titik yang diletakkan secara simetris terhadap origin, maka dapat didefinisikan bidang batas A dengan metode berikut.

$$boundary = A - (A \ominus B) \quad (2\text{-}4)$$

Pada gambar 2.14, citra A terkena erosi oleh SE B. Selanjutnya terjadi operasi pengurangan citra A terhadap citra hasil erosi.



Gambar 2. 14 Boundary detection

(**Sumber:** Rafael C. G. and Richards E. W., 2010)

2.6 Ekstraksi Ciri

Ekstraksi ciri merupakan pengambilan ciri pada objek melalui proses tertentu. Ciri atau fitur merupakan karakteristik unik dari suatu objek. Karakteristik fitur yang baik sebisa mungkin memenuhi persyaratan beikut.

a. Dapat membedakan suatu objek dengan yang lainnya.
b. Memperhatikan kompleksitas komputasi dalam memperoleh fitur. Kompleksitas yang tinggi akan menjadi beban tersendiri dalam menemukan fitur.
c. Tidak terikat dalam arti bersifat invarian terhadap berbagai transformasi.
d. Jumlahnya sedikit.

Ekstraksi fitur dikatagorikan menjadi 3 level, yaitu level rendah (*low-level*), level medium (*middle-level*), dan level tinggi (*high-level*). *Low-level* fitur merupakan ekstraksi ciri berdasarkan fitur visual seperti warna dan tekstur. *Middle-level* fitur merupakan ekstraksi berdasarkan wilayah citra yang ditentukan dengan segmentasi. High-level fitur merupakan ekstraksi ciri berdasarkan informasi semantik yang terkandung dalam citra. Untuk memperoleh suatu fitur atau ciri dari citra diperlukan metode analisis citra. Berbagai metode ekstraksi fitur suatu citra dijelaskan sebagai berikut.

2.6.1 Amplitudo

Ciri amplitudo merupakan ciri yang paling sederhana dan paling berguna dari suatu objek. Ciri amplitudo berhubungan dengan properti fisik suatu objek. Ciri amplitudo dapat ditentukan pada suatu pixel citra tertentu seperti amplitudo $f_{j,k}$ pada pixel berkoordinat (j,k) atau meliputi suatu pixel tetangga dengan ukuran tertentu yang berpusat pada koordinat (j,k). Ciri amplitudo yang meliputi pixel tetangga berukuran WxW dapat berupa:

$$Rata - rata = m_{j,k} = \frac{1}{W^2}\sum_{m=-w}^{w}\sum_{n=-w}^{w} f_{j+m,k+n} \quad (2\text{-}5)$$

$$Median = s_{j,k} = \frac{1}{W^2}\left[\sum_{m=-W}^{W}\sum_{n=-W}^{W}\left[f_{j+m,k+n} - m_{j+m,k+n}\right]^2\right]^{\frac{1}{2}} \quad (2\text{-}6)$$

2.6.2 Histogram

Ciri histogram didasarkan pada histogram dari suatu citra. Bila x menyatakan tingkat keabuan pada suatu citra maka probabilitas dari x dinyatakan dengan:

$$P(x) = \frac{Banyaknya\ titik{-}titik\ yang\ memiliki\ tingkat\ keabuan\ x}{Total\ banyaknya\ titik\ pada\ daerah\ suatu\ citra} \quad (2\text{-}7)$$

2.6.3 Matriks co-occurance

Matriks co-occurance adalah matriks yang dibangun menggunakan histogram tingkat kedua. Matriks co-occurance merupakan matriks berukuran LxL (L menyatakan tingkat keabuan) dengan elemen-elemen $P(x_1, x_2)$ yang merupakan distribusi probabilitas bersama dari pasangan pixel dengan tingkat keabuan x_1 yang berlokasi pada koordinat (j,k) dengan x_2 yang berlokasi pada koordinat (m,n). Koordinat pasangan pixel tersebut berjarak r dengan sudut θ. Histogram tingkat kedua $P(x_1, x_2)$ dihitung dengan pendekatan sebagai berikut.

$$P(x) = \frac{Banyaknya\ titik{-}titik\ yang\ memiliki\ tingkat\ keabuan\ x_1\ dan\ x_2}{Total\ banyaknya\ titik\ pada\ daerah\ suatu\ citra} \quad (2\text{-}8)$$

2.6.4 Gradient

Nilai gradient untuk setiap pixel pada daerah suatu citra dapat dihitung dengan rumus berikut.

$$G(x,y) = \sqrt{\left(f_{x+1,y} - f_{x-1,y}\right)^2 + \left(f_{x,y+1} - f_{x,y-1}\right)^2} \quad (2\text{-}9)$$

Dengan $f_{x,y}$ adalah nilai pixel pada posisi (x, y). Fitur yang dapat dihitung berdasarkan nilai gradient, antara lain *GrMean*, *GrVariance*, *GrSkewnes*, *GrKurtosis*.

2.6.5 Deteksi tepi

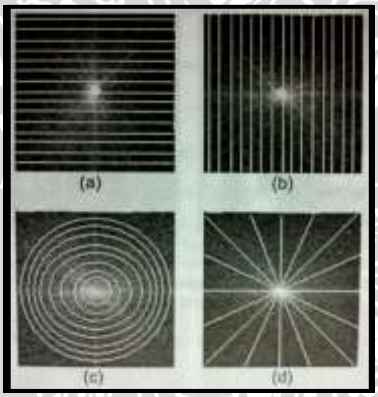
Banyaknya tepi pada pixel tetangga dapat diukur sebagai suatu fitur tekstur. Untuk mengukur ciri tekstur ini, suatu citra harus mengalami proses deteksi tepi. Citra deteksi tepi dapat diperoleh dengan menggunakan metode deteksi tepi seperti, operator robert, sobel prewit, krisch dan canny. Ciri tekstur pada suatu daerah citra yang berukuran WxW didefinisikan sebagai berikut.

$$T_{j,k} = \frac{1}{W^2} \sum_{m=-w}^{w} \sum_{n=-w}^{w} E_{j+m,k+n} \quad (2\text{-}10)$$

$E_{j,k}$ merupakan citra hasil proses deteksi tepi.

2.6.6 Spektrum fourier

Transformasi fourier membawa suatu citra dari domain spasial f(x,y) ke domain frekuensi F(p,q). Beberapa fitur penting yang dapat diperoleh dari spektrum fourier adalah celah horisontal, celah vertikal, cincin, dan sektor seperti yang ditunjukkan gambar 2.15.



Gambar 2. 15 Ciri spektrum Fourier

(a) celah horisontal (b) celah vertikal (c) cincin (d) sektor

(**Sumber:** Darma Putra, 2010)

2.6.7 Wavelet

Beberapa metode untuk memperoleh fitur suatu citra berdasarkan wavelet adalah fitur energi subband dan fitur berdasarkan pemilihan koefisien wavelet. Fitur energi subband diperoleh dengan menghitung energi yang terkandung pada setiap subband yaitu komponen aproksimasi dan komponen detil horisontal, vertikal, dan diagonal. Pada setiap skala, suatu citra hasil transformasi wavelet terbagi atas 4 subband. Fitur subband dapat dihitung menggunakan rumus sebagai berikut.

$$E_{subband\_scale} = \frac{\sum_{x,y}(d_{x,y}^{subband})^2}{n} \quad (2\text{-}11)$$

Dengan n menyatakan jumlah pixel pada setiap subband. Fitur berdasarkan pemilihan koefisien wavelet dapat didasarkan pada 2 cara berikut. Pertama memilih sejumlah kecil koefisien berdasarkan magnitudo terbesar tanpa memperhatikan subband. Kedua, memilih sejumlah kecil koefisien berdasarkan magnitudo terbesar dengan memperhatikan subband.

2.6.8 Fitur berdasarkan warna

Metode yang dapat digunakan untuk mendapatkan ciri berdasarkan warna adalah dengan menggunakan *mean* warna dan histogram warna. *Mean* warna adalah nilai rata-rata kanal dari suatu objek. *Mean* dihitung pada setiap piksel untuk masing-masing nilai *kanal* piksel. Berikut ini merupakan mean warna kanal RGB.

$$r_{avg} = \frac{\sum_{i=1}^{N} r(i)}{N}; g_{avg} = \frac{\sum_{i=1}^{N} g(i)}{N}; b_{avg} = \frac{\sum_{i=1}^{N} b(i)}{N} \quad (2\text{-}12)$$

dengan:

r_{avg}, g_{avg}, b_{avg} = rata-rata intensitas warna R, G, B

r(i), g(i), b(i) = nilai intensitas warna R, G, B pada suatu pixel

N = total pixel pada objek.

Histogram warna merupakan fitur yang juga dapat digunakan untuk merepresentasikan ciri warna suatu citra. Citra pada umumnya dikonversi ke dalam suatu ruang warna tertentu, kemudian setiap komponen ruang warna dibuat histogramnya.

2.6.9 Tapis gabor

Tapis gabor merupakan metode yang handal untuk menghasilkan fitur dalam hal biometrika seperti fingercode dan iriscode. Pada tapis gabor 2D terdapat beberapa parameter yang berpengaruh terhadap hasil ekstraksi fitur antara lain ukuran kernel, parameter θ yang mengontrol orientasi dari tapis gabor, parameter μ, dan parameter standar deviasi yang menyesuaikan dengan ukuran tapis.

2.6.10 Analisis Bentuk Atribut Geometri

Salah satu obyek yang memiliki bentuk yang sangat sederhana adalah lingkaran. Di samping sederhana, lingkaran memiliki bentuk yang sangat simetris. Seperti diketahui, luas dan keliling sebuah lingkaran dapat diketahui dengan:

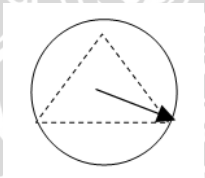
$$A = \pi R^2 \quad (2\text{-}13)$$

$$P = 2\pi R = \pi D \quad (2\text{-}14)$$

dimana R adalah jari-jari dan D merupakan garis tengah lingkaran. Komputasi untuk kedua properti lingkaran ini dapat dilakukan pada region-region citra, dan ini merupakan bentuk dasar dari ukuran kebundaran(circularity). Rasio P^2 / A untuk sebuah lingkaran adalah 4π, dan ini merupakan nilai minimum untuk setiap region. Dari sini, dapat dirumuskan ukuran kebundaran obyek adalah:

$$C = \frac{P^2}{4\pi A} \quad (2\text{-}15)$$

C memiliki nilai minimum 1 untuk lingkaran dan terus menaik untuk region yang lebih kompleks.



Gambar 2. 16 Bounding Circle

(**Sumber:** Lukman Talibo, 2004)

Untuk mendapatkan nilai *bounding circle* dilakukan pendekatan dengan memberikan nilai maksimum 1 untuk lingkaran dan nilai yang terus menurun untuk obyek yang lebih kompleks. Nilai ini dapat mewakili *bounding circle*. Perbandingan antara luas suatu lingkaran dengan luas lingkaran terkecil yang dapat menutupi lingkaran tersebut adalah 1.

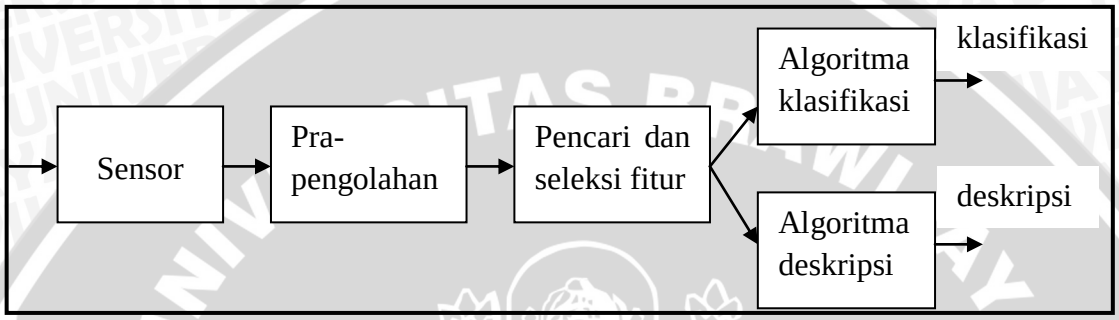
$$C = \frac{1}{\left(\frac{P^2}{4\pi A}\right)} = \frac{4\pi A}{P^2} \quad (2.16)$$

2.7 Pengenalan Pola

Teknik pengenalan pola merupakan salah satu komponen penting dari mesin atau sistem cerdas yang digunakan baik untuk mengolah data maupun dalam

pengambilan keputusan. Secara umum pengenalan pola adalah suatu ilmu untuk mengklasifikasikan atau menggambarkan sesuatu berdasarkan pengukuran kuantitatif ciri atau sifat utama dari suatu objek. Pola adalah suatu entitas yang terdefinisi dan dapat diidentifikasikan serta diberi nama. Pola bisa merupakan kumpulan hasil pengukuran atau pemantauan dan bisa dinyatakan dalam notasi vektor atau matriks.

Stuktur dari sistem pengenalan pola terdiri dari sensor, algoritma atau mekanisme pencari fitur, dan algoritma untuk klasifikasi atau pengenalan seperti yang ditunjukkan oleh gambar 2.17.



Gambar 2. 17 Struktur sistem pengenalan pola

(**Sumber:** Darma Putra, 2010)

Sensor berfungsi untuk menangkap objek dan mengubah menjadi sinyal digital melalu digitalisasi. Pra-pengolahan berfungsi untuk mempersiapkan citra agar dapat menghasilkan ciri yang baik. Pencari dan seleksi fitur berfungsi untuk menemukan karakteristik pembeda yang mewakili sifat utama sinyal dan sekaligus mengurangi dimensi sinyal menjadi sekumpulan bilangan yang sedikit tetapi representatif. Algoritma klasifikasi berfungsi untuk mengelompokkan fitur ke dalam kelas yang sesuai. Algoritma deskripsi berfungsi memberikan deskripsi pada sinyal.

Fitur atau disebut juga atribut adalah semua hasil pengukuran yang bisa diperoleh dan merupakan karakteristik pembeda dari objek fitur. Fitur dapat berupa simbol seperti warna, numerik seperti berat, atau gabungan keduanya. Fitur dapat dinyatakan dengan variabel kontinyu, diskret, atau diskret-biner.

Vektor fitur adalah gabungan atau kombinasi dari beberapa fitur dan dinyatakan sebagai vektor kolom. Ruang fitur adalah ruang yang dibentuk oleh vektor fitur dan merupakan cara untuk memvisualisasikan distribusi vektor fitur. Scatter plot merupakan alat visualisasi untuk menentukan distribusi dari distribusi vektor bila dimensi vektor kurang dari sama dengan 3. Pola dapat dikatakan sama dengan fitur atau vektor fitur yang merupakan sifat utama dari suatu objek. Namun dalam pengenalan

(klasifikasi), pola merupakan sepasang variabel (x, ω) dengan x menyatakan sekumpulan pengamatan atau fitur atau vektor fitur dan ω merupakan konsep dibalik pengamatan.

Pemilah merupakan teknik atau metode untuk mengelompokkan vektor fitur ke dalam kelas-kelas tertentu. Vektor fitur untuk pelatihan merupakan vektor fitur yang telah diketahui kelasnya dan digunakan untuk merancang pemilah.

2.8 Euclidean Distance

Jarak digunakan untuk menentukan tingkat kesamaan atau ketidaksamaan dua vektor fitur. Tingkat kesamaan berupa suatu nilai dan berdasarkan nilai tersebut dua vektor fitur akan dikatakan mirip atau tidak. Salah satu metode menghitung jarak dua vektor fitur yaitu *euclidean distance*.

Euclidean distance adalah pengukuran yang sering digunakan untuk menghitung kesamaan 2 vektor. *Euclidean distance* menghitung akar dari kuadrat dua vektor. Rumusnya adalah sebagai berikut:

$$d_{ij} = \sqrt{\sum_{k=1}^{n} \left(x_{ik} - x_{jk}\right)^2}$$

$$= \sqrt{\left(x_{i1} - x_{j1}\right)^2 + \left(x_{i2} - x_{j2}\right)^2 + \left(x_{i3} - x_{j3}\right)^2 + \cdots + \left(x_{in} - x_{jn}\right)^2} \qquad (2\text{-}17)$$

dengan:

d_{ij} = euclidean distance vektor i dan j

k = jumlah variabel vektor (1,2,3,...,n)

x_{ik} = nilai variabel vektor i

x_{jk} = nilai variabel vektor j

2.9 Jaringan Saraf Tiruan

Jaringan saraf tiruan atau *Artificial Neural Network* yang sering disingkat dengan ANN adalah sistem pemroses informasi yang memiliki karakteristik kinerja tertentu yang sama dengan jaringan saraf biologis. ANN telah dikembangkan sebagai generalisasi model matematika suatu saraf biologis manusia, berdasarkan asumsi bahwa:

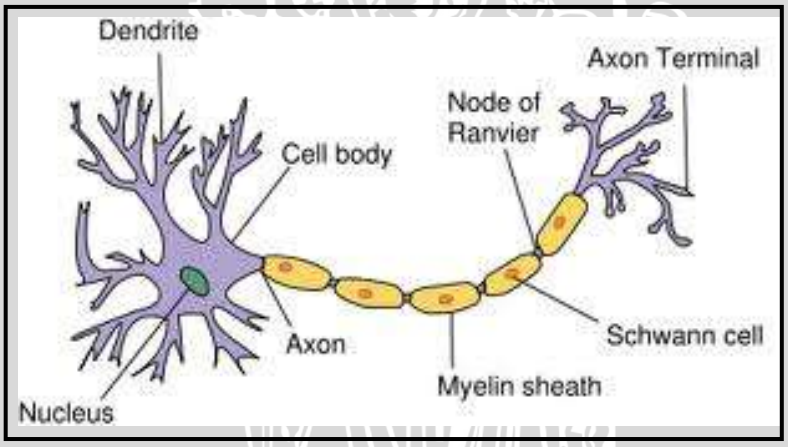
- Pengolahan informasi terjadi pada elemen sederhana yang disebut neuron.
- Sinyal antar neuron dilewatkan melalui suatu hubungan.

- Setiap hubungan memiliki berat yang terkait, yang dalam jaringan saraf, mengalikan sinyal yang ditransmisikan.
- Setiap neuron menerapkan fungsi aktivasi terhadap input jaringan untuk menentukan sinyal outputnya.

Sebuah jaringan saraf ditandai oleh (1) polanya koneksi antara neuron (disebut arsitektur), (2) metode menentukan bobot (disebut pelatihan, atau belajar, algoritma), dan (3) fungsi aktivasi.

2.9.1 Model Neuron Biologis

Pada dasarnya jaringan saraf atau *neural network* terdiri atas banyak elemen pemroses sederhana yang disebut neuron, sel unit, atau simpul. Sebagai bahan perbandingan, otak seekor cacing diperkirakan memilikki 1.000 neuron dan otak manusia memiliki sekitar 100 miliar. Setiap sel saraf berhubungan dengan sel saraf lainnya memakai saluran komunikasi yang teratur dengan suatu bobot penghubung. Gambar 2.18 berikut menunjukkan gambar struktur neuron pada otak manusia.



Gambar 2. 18 Struktur neuron
(**Sumber:** Darma Putra, 2010)

2.9.2 Model dan konsep dasar ANN

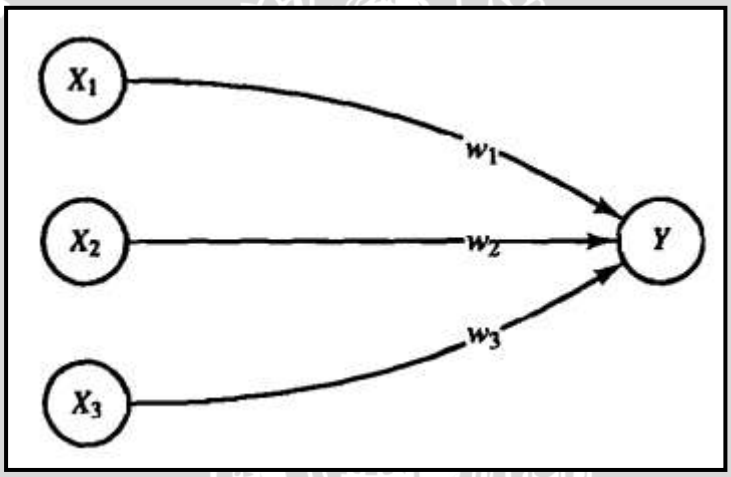
Sebuah jaringan saraf terdiri dari sejumlah elemen pemroses sederhana yang disebut neuron, unit, sel, atau node. Setiap neuron terhubung ke neuron lain melalui jaringan komunikasi terarah, masing-masing dengan bobot yang terkait. Bobot tersebut merupakan informasi yang digunakan oleh jaringan untuk memecahkan masalah. Jaringan saraf dapat diterapkan pada berbagai macam masalah, seperti menyimpan dan mengingat data atau pola, mengklasifikasi pola, melakukan pemetaan umum dari pola

input ke pola output, pengelompokan pola yang sama, atau mencari solusi bagi permasalahan optimasi yang dibatasi.

Setiap neuron memiliki keadaan internal, disebut aktivasi atau tingkat aktivitas, yang mana fungsi dari input telah diterima. Biasanya, neuron mengirimkan aktivasi sebagai sinyal ke neuron lainnya. Penting untuk dicatat bahwa neuron dapat mengirim hanya satu sinyal pada satu waktu, meskipun sinyal tersebut disebarkan ke neuron lainnya.

Sebagai contoh, suatu neuron Y, diilustrasikan pada Gambar 2.19, menerima masukan dari neuron X_1, X_2, dan X_3. Aktivasi (sinyal output) dari neuron tersebut masing-masing x_1, x_2, dan x_3. Bobot pada koneksi dari X_1, X_2, dan X_3 ke neuron Y masing-masing w_1 untuk w_2, dan w_3. Input jaringan y_in ke neuron Y adalah jumlah dari sinyal berbobot dari neuron X_1, X_2, dan X_3 yaitu,

$$y_{in} = w_1 x_1 + w_2 x_2 + w_3 x_3$$



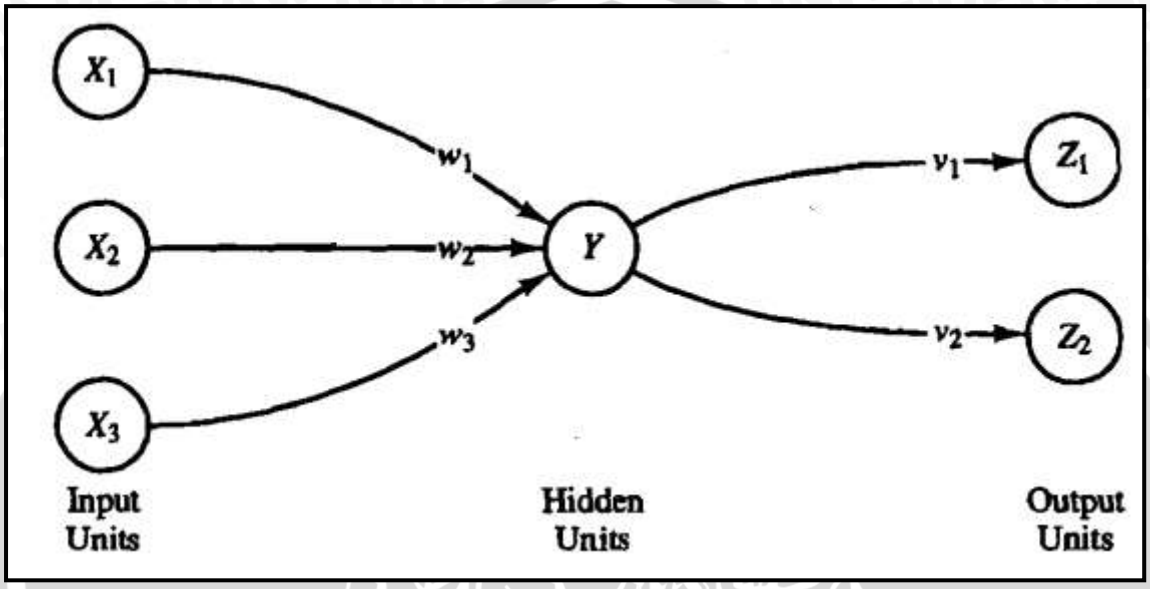
Gambar 2. 19 Neuron sederhana

(**Sumber:** Laurene Fausett, 1993)

Aktivasi y dari neuron Y diberikan oleh beberapa fungsi jaringan input tersebut y=f(y_in), misalnya fungsi sigmoid atau salah satu dari sejumlah fungsi aktivasi lainnya.

Sekarang anggaplah lebih lanjut bahwa neuron Y terhubung ke neuron Z_1 dan Z_2, dengan masing-masing bobot v_1 dan v_2, seperti yang ditunjukkan pada Gambar 2.20. Neuron Y mengirimkan sinyal y untuk masing-masing unit. Namun, secara umum, nilai-nilai yang diterima oleh neuron Z_1 dan Z_2 akan berbeda, karena setiap sinyal diskalakan oleh bobot v_1 dan v_2. Dalam jaringan yang khas, aktivasi $z_1$dan z_2 neuron Z_1 dan Z_2 akan tergantung pada masukan dari beberapa neuron.

Meskipun jaringan saraf pada gambar 2.20 adalah sangat sederhana, namun adanya unit tersembunyi, bersama dengan fungsi aktivasi non-linear, memberikan kemampuan untuk memecahkan banyak masalah yang lebih baik daripada yang dapat diselesaikan dengan jaringan hanya dengan input dan output unit. Di sisi lain, lebih sulit untuk melatih (yaitu, menemukan nilai-nilai optimal untuk bobot) jaringan dengan unit tersembunyi.



Gambar 2. 20 Jaringan saraf tiruan sederhana

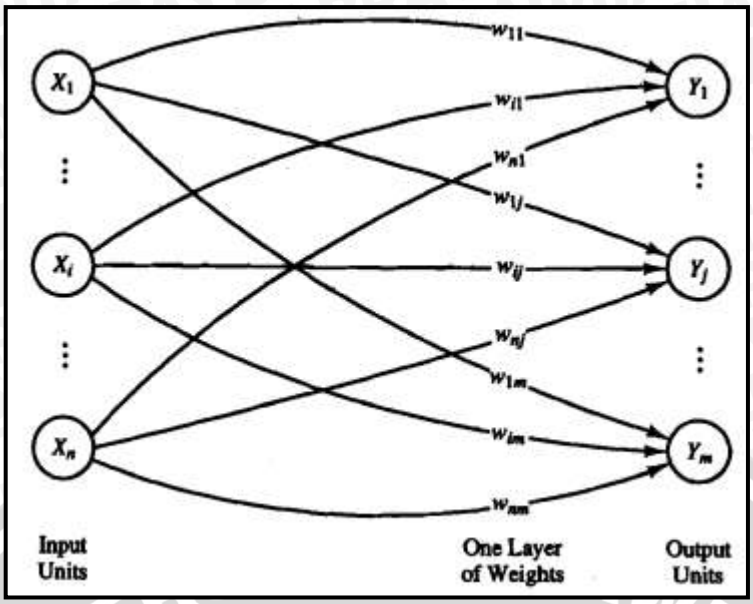
(**Sumber:** Laurene Fausett, 1993)

2.9.3 Arsitektur ANN

ANN memiliki beberapa arsitektur yang sering dipakai dalam berbagai aplikasi. Arsitektur tersebut antara lain:

2.9.3.1 Jaringan lapisan tunggal (*single layer network*)

Jaringan dengan lapisan tunggal terdiri dari 1 *layer* input dan 1 *layer* output. Setiap neuron/unit yang terdapat di dalam lapisan input selalu terhubung dengan setiap neuron yang terdapat pada layer output. Jaringan ini hanya menerima input kemudian secara langsung akan mengolahnya menjadi output tanpa harus melalui lapisan tersembunyi.

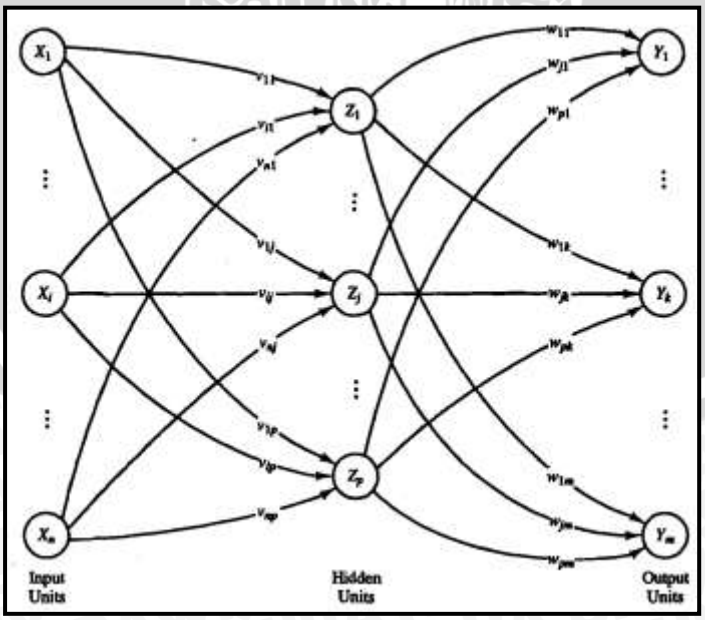


Gambar 2. 21 Arsitektur lapisan tunggal

(**Sumber:** Laurene Fausett, 1993)

2.9.3.2 Jaringan lapisan jamak (*multi layer network*)

Jaringan dengan lapisan jamak memiliki ciri khas tertentu yaitu memiliki 3 jenis *layer* yakni *layer* input, *layer* output dan *layer* tersembunyi. Jaringan dengan lapisan banyak ini dapat menyelesaikan permasalahan yang lebih kompleks dibandingkan dengan jaringan lapisan tunggal. Namun proses pelatihan cenderung membutuhkan waktu yang lebih lama.

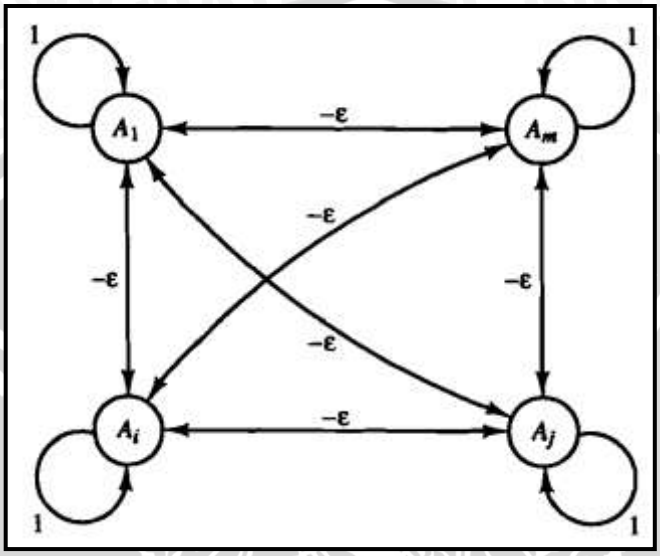


Gambar 2. 22 Arsitektur lapisan jamak

(**Sumber:** Laurene Fausett, 1993)

2.9.3.3 Jaringan dengan lapisan kompetitif (*competitive layer network*)

Pada jaringan ini, sekumpulan neuron bersaing untuk mendapatkan hak menjadi aktif. Lapisan kompetitif merupakan bagian dari sejumlah jaringan saraf yang besar. Biasanya, interkoneksi antara neuron pada lapisan kompetitif tidak ditampilkan dalam arsitektur diagram untuk jaringan tersebut.



Gambar 2. 23 Arsitektur lapisan kompetitif

(**Sumber:** Laurene Fausett, 1993)

2.9.4 Pengaturan bobot

Selain arsitektur, metode pengaturan nilai-nilai bobot (pelatihan) adalah ciri khas penting dari suatu jaringan saraf tiruan. Terdapat dua jenis pelatihan yaitu pelatihan yang diawasi dan tidak diawasi, di samping itu ada jaringan yang bobotnya adalah tetap tanpa proses berulang-ulang pelatihan. Klasifikasi dan asosiasi pola dapat dianggap sebagai bentuk khusus dari permasalahan umum pemetaan pola atau vektor input ke pola atau vector output yang lebih spesifik.

a. *Supervised learning* (pembelajaran terawasi)

Pada metode ini, setiap pola yang diberikan kedalam ANN telah diketahui outputnya. Selisih antara pola output aktual (output yang dihasilkan) dengan pola output target (output yang dikehendaki) yang disebut eror digunakan untuk mengoreksi bobot ANN sehingga ANN mampu untuk menghasilkan output sedekat mungkin dengan pola target yang telah diketahui oleh ANN.

b. *Unsupervised learning* (pembelajaran tak terawasi)

Pada metode ini tidak memerlukan target output. Pada metode ini tidak dapat ditentukan hasil seperti apakah yang diharapkan selama proses pembelajaran. Selama proses pembelajaran, nilai bobot disusun dalam *range* tertentu tergantung pada nilai input yang diberikan. Tujuan dari pembelajran ini adalah mengelompokkan unit-unit yang hampir sama dalam suatu area tertentu. Pembelajaran ini biasanya sangat cocok dalam klasifikasi pola.

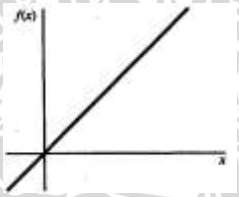
2.9.5 Fungsi aktivasi umum

Seperti disebutkan sebelumnya, operasi dasar dari sebuah neuron buatan melibatkan penjumlahan sinyal input berbobot dan menerapkan output, atau aktivasi, fungsi.

a. Fungsi identitas (linier)

$$f(x) = x \; for \; all \; x$$

Fungsi liner memiliki nilai output yang sama dengan nilai inputnya.



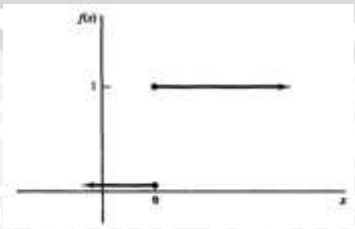
Gambar 2. 24 Fungsi identitas

(**Sumber:** Laurene Fausett, 1993)

b. Fungsi biner dengan threshold θ

$$f(x) = \begin{cases} 1 & if \; x \geq \theta \\ 0 & if \; x < \theta \end{cases}$$

Jaringan layer tunggal sering menggunakan fungsi step untuk mengkonversi input, yang mana nilai variabelnya *continue*, ke unit output yang merupakan sinyal biner (1 atau 0) atau bipolar (1 atau - 1). Fungsi biner juga dikenal sebagai fungsi *threshold*.



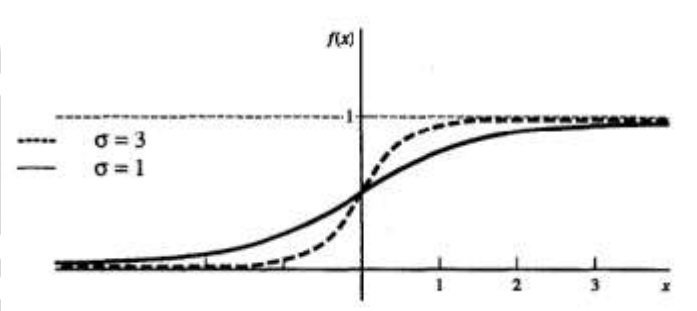
Gambar 2. 25 Fungsi threshold

(**Sumber:** Laurene Fausett, 1993)

c. Fungsi sigmoid biner

Fungsi ini merupakan fungsi aktivasi yang sangat berguna dan digunakan dalam jaringan syaraf yang dilatih oleh backpropagation. Fungsi sigmoid biner memiliki nilai pada range 0 sampai 1. Oleh karena itu fungsi ini sering digunakan untuk jaringan saraf tiruan yang membutuhkan nilai output yang terletak pada interval 0 dan 1. Namun fungsi ini bisa juga digunakan oleh jaringan saraf tiruan yang nilai outputnya 0 dan 1.

$$f(x) = \left\{\frac{1}{1 + \exp(-\sigma x)}\right\}$$



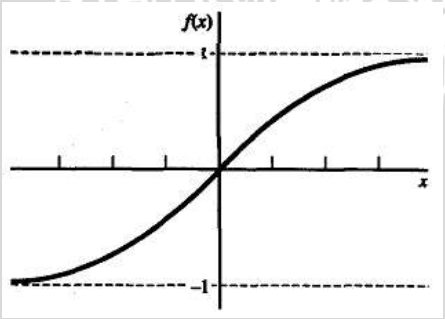
Gambar 2. 26 Fungsi sigmoid biner

(**Sumber:** Laurene Fausett, 1993)

d. Fungsi sigmoid bipolar

$$f(x) = \left\{\frac{1 - \exp(-\sigma x)}{1 + \exp(-\sigma x)}\right\}$$

Fungsi sigmoid bipolar hampir sama dengan fungsi sigmoid biner, hanya saja output dari fungsi ini memiliki range antara -1 sampai 1.



Gambar 2. 27 Fungsi sigmoid biner

(**Sumber:** Laurene Fausett, 1993)

2.9.6 *Learning rate*

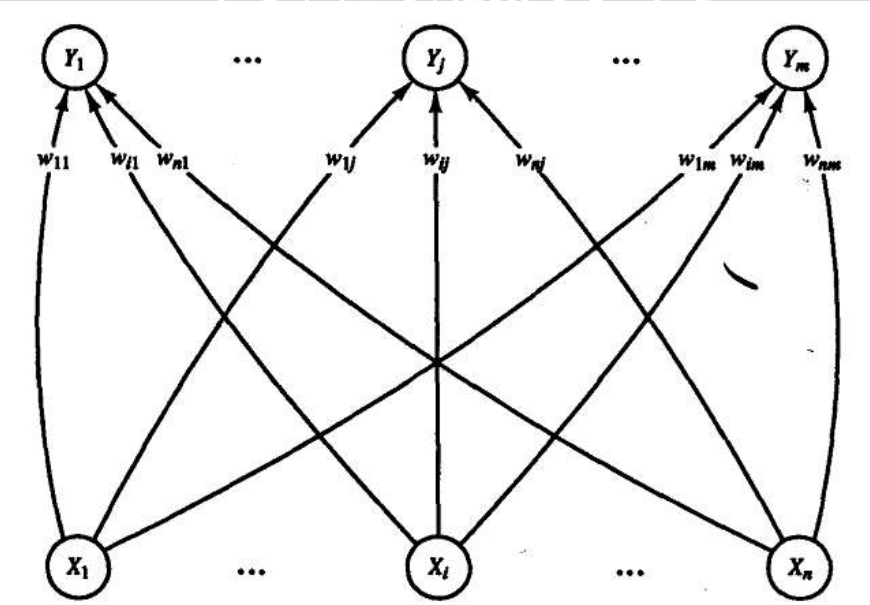
Dalam pemilihan learning rate terdapat beberapa pertimbangan. Pelatihan yang lambat membutuhkan waktu yang lebih lama untuk menghasilkan sistem yang terlatih. Pelatihan yang cepat memungkan waktu yang singkat akan tetapi jaringan belum tentu

menghasilkan sistem yang terlatih seperti pada pelatihan yang lambat. Fungsi pelatihan memiliki beberapa ketentuan untuk nilai learning rate. Secara umum bernilai antara 0 sampai positif 1. Jika lebih dari 1 algoritma pelatihan akan mengalami *overshoot* dalam perubahan bobot. Nilai learning rate yang kecil tidak akan memperbaiki kesalahan dengan cepat, akan tetapi memiliki kesempatan untuk mencapai konvergensi minimum terbaik.

2.9.7 *Learning Vector Quantization* (LVQ)

Learning Vector Quantization (LVQ) [Kohonen, 1989a, 1990a] adalah sebuah metode klasifikasi dimana setiap unit output mewakili suatu kelas atau kategori. Vektor bobot suatu unit output sering disebut sebagai vektor referensi suatu kelas pada suatu unit. Selama pelatihan, unit keluaran diposisikan dengan menyesuaikan bobot mereka melalui pelatihan diawasi.

LVQ adalah salah satu jaringan saraf tiruan yang merupakan algoritma pembelajaran kompetitif terawasi versi dari algoritma *Kohonen Self-Organizing Map* (SOM). Tujuan dari algoritma ini adalah untuk mendekati distribusi kelas vektor untuk meminimalkan kesalahan dalam pengklasifikasian. Arsitektur jaringan LVQ ini terdiri dari input, lapisan kohonen, dan lapisan output, seperti pada gambar 2.22.



Gambar 2. 28 Arsitektur jaringan LVQ

(**Sumber:** Laurene Fausett, 1993)

Algoritma jaringan LVQ mencari unit output yang terdekat dengan unit input. Jika x dan w berada pada kelas yang sama, maka bobot vektor dipindahkan mendekati input vektor yang baru. Jika x dan w merupakan kelas yang berbeda, maka bobot vektor dipindahkan menjauhi input vektor. Algoritma LVQ adalah sebagai berikut:

Langkah 0. Inisialisasi bobot awal, *learning rate*, $\alpha(0)$.

Langkah 1. Selama kondisi berhenti adalah salah, kerjakan langkah 2-6.

Langkah 2. Untuk setiap input pelatihan vektor x, lakukan langkah 3-4

Langkah 3. Temukan J agar $\| x-w_J \|$ adalah minimum

Langkah 4. Perbaiki w_J dengan ketentuan:

Jika $T=C_J$ maka:

$w_J(\text{baru}) = w_J(\text{lama}) + \alpha[x-w_J(\text{lama})]$

Jika $T \neq Cj$ maka:

$w_J(\text{baru}) = w_J(\text{lama}) - \alpha[x-w_J(\text{lama})]$

Langkah 5. Kurangi *learning rate*.

Langkah 6. Uji kondisi berhenti:

Kondisi ini dapat ditentukan dengan menetapkan jumlah eksekusi langkah 1 atau saat learning rate mencapai nilai yang cukup kecil.

Keterangan:

x	vektor pelatihan(x_1,..., x_i).
T	kategori atau kelas yang tepat untuk vektor pelatihan.
w_J	bobot vektor untuk unit output J($w_{1J,...}$, $w_{iJ,...}$, w_{nJ}).
C_J	kategori atau kelas yang diwakili oleh unit output J.
$\| x-w_J \|$	Euclidean distance antara vektor input dan bobot vektor untuk unit output ke-J

Terdapat beberapa metode dalam penentuan bobot awal dalam LVQ. Metode pertama adalah mengambil data vektor pelatihan pertama tiap kelas kemudian menggunakannya sebagai bobot vektor. Metode kedua adalah menggunakan K-means clustering dan ketiga adalah menggunakan *self-organizing map*.

2.10 Basis Data

2.10.1 Pengertian Basis Data

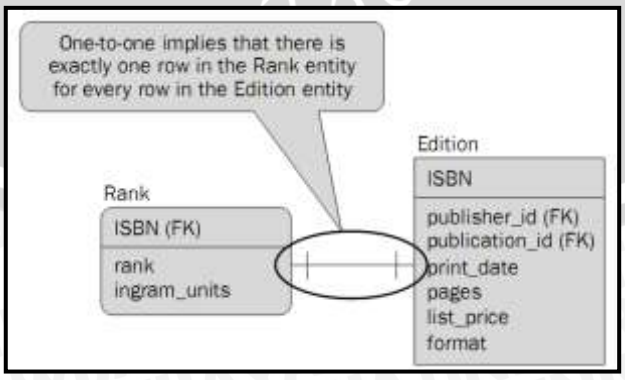
Basis data adalah sebuah cara mendokumentasikan berbagai macam data yang kemudian dimanajemen dengan sebuah sistem untuk kemudian disimpan dalam sebuah media penyimpanan. Dengan demikian data-data tersebut dapat diakses dengan mudah dan cepat. Media penyimpanan tersebut dapat kita ibaratkan sebuah *storage* penyimpanan, misalnya hardisk. Dalam basisdata, data yang ada tidak hanya diletakkan dan disimpan begitu saja dalam sebuah media penyimpanan, akan tetapi dikelola dengan sebuah sistem pengaturan basisdata yang seiring disebut dengan *Database Management System (DBMS)*. Dengan begitu, suatu data dengan jumlah besar dan kompleks dapat tersusun sangat baik sehingga memungkinkan pengaksesan data dengan mudah dan cepatoleh pengguna. Basisdata merupakan sekumpulan informasi yang sangat kompleks yang berguna untuk mengatur semua data yang ada di dalamnya sehingga dapat diakses oleh pengguna dengan mudah dan cepat.

2.10.2 Relasional Database Model

Relational Database Model merupakan salah satu jenis basis data yang paling banyak digunakan. Pada model ini, tabel-tabel dapat dihubungkan satu sama lain sehingga timbul suatu relasi antar tabel. Ada berbagai macam relasi antar tabel, diantaranya:

1). *One-to-One*

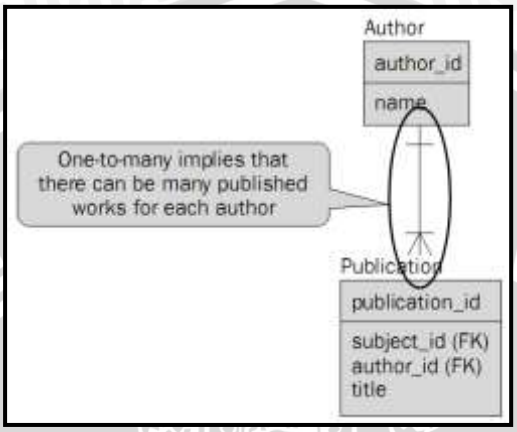
Relasi *One-to-One* biasanya digunakan untuk menghilangkan nilai NULL pada baris kolom tabel. Satu baris tabel mengacu pada satu baris pada tabel lainnya. Relasi *One-to-one* ditujukkan pada gambar 2.23 sebagai berikut.



Gambar 2. 29 Contoh Relasi *One to One*
(**Sumber:** Gavin Powell, 2006)

2). *One-to-Many*

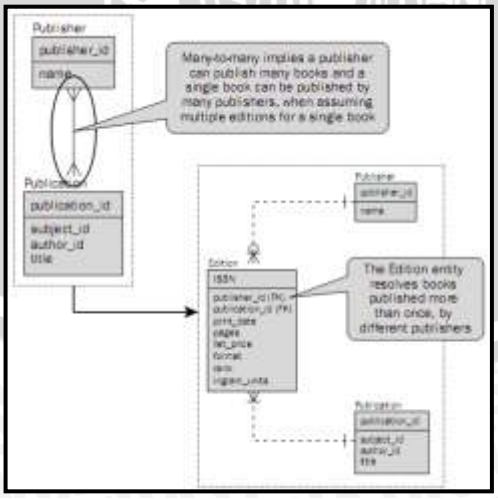
Relasi *One-to-Many* merupakan relasi yang paling sering dijumpai pada *relational database model*. Satu baris tabel mengacu pada satu baris atau lebih pada tabel lainnya. Relasi *One-to-Many* ditunjukkan pada gambar 2.24 sebagai berikut.



Gambar 2. 30 Contoh Relasi *One to Many*
(**Sumber:** Gavin Powell, 2006)

3) *Many-to-Many*

Relasi *many-to-many* berarti bahwa setiap baris tabel memiliki kemungkinan memiliki relasi dengan banyak baris pada tabel lainnya, dan sebaliknya. Biasanya relasi *many-to-many* membutuhkan tabel tambahan sebagai penghubung. Relasi *many-to-many* ditunjukkan pada gambar 2.25 sebagai berikut.



Gambar 2. 31 Contoh Relasi *Many to Many*
(**Sumber:** Gavin Powell, 2006)

BAB III
METODE PENELITIAN

Pada tahap ini dijelaskan mengenai langkah-langkah yang akan dilakukan untuk merancang dan mengimplementasikan perangkat lunak yang akan dibuat. Adapun langkah-langkah yang akan dilakukan adalah sebagai berikut:

3.1 Studi Literatur

Dalam penyusunan skripsi ini, pengumpulan data dilakukan dengan melakukan studi literatur dengan sasaran tinjauan antara lain :

1) Informasi internet.
2) Pustaka-pustaka referensi.
3) Pustaka penunjang.

Studi literatur yang dilakukan bertujuan untuk mengkaji hal-hal yang berhubungan dengan teori-teori yang mendukung dalam perencanaan dan perealisasian aplikasi. Adapun teori-teori yang dikaji adalah sebagai berikut:

1) Buah
2) Pengolahan citra digital
3) Ekstraksi ciri
4) Algoritma *Learning Vector Quantization*
5) Pemrograman visual *C#*

3.2 Penentuan Spesifikasi Aplikasi

Menentukan perangkat yang digunakan untuk menunjang pembuatan aplikasi:

1. Perangkat Keras:
   a. Laptop

      AMD Turion™ II X2 *processor* M520 2.3GHz, *mainboard* Aspire 4540, DDR II 3072MB, VGA Ati Radeon HD 4200 256MB, harddisk 320 GB

   b. *Webcam*

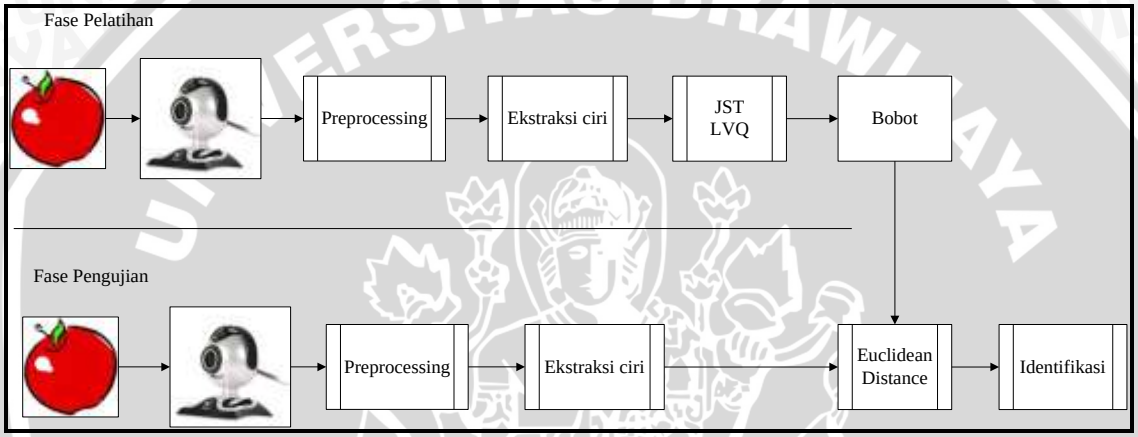
      i-Tech, Resolusi: 1,3 megapiksel, *frame rate*: up to 30fps, mic dan *snapshot button*: *built in*, antarmuka: USB 2.0

   c. Lightent

2. Perangkat Lunak:
   a. Windows 7 Ultimate
   b. Microsoft Visual Studio 2010 .NET

3.3 Perancangan dan Implementasi Sistem

Pada aplikasi identifikasi jenis buah berdasarkan warna dan bentuk menggunakan webcam ini dibagi menjadi beberapa tahap perancangan, yaitu perancangan basis data, inisialisasi *webcam*, pengambilan gambar, *preprocessing*, ekstraksi ciri, pembelajaran *Learning Vector Quantization(LVQ)*, identifikasi buah.



Gambar 3. 1 Diagram sistem

Pada gambar 3.1 dijelaskan aplikasi berjalan dalam 2 fase yaitu fase pelatihan dan fase pengujian. Kedua fase ini memiliki masukan berupa citra buah yang ditangkap oleh *webcam* sehingga perlu dilakukan inisialisasi *webcam* agar aplikasi dapat mengambil gambar *online*. Setelah dilakukan pengambilan gambar maka citra buah dipisahkan dengan *background* yang telah ditetapkan. Buah yang telah terpisah dengan *background* di ekstraksi cirinya. Ciri yang diambil dari buah adalah warna dan bentuk. Warna dalam hal ini meliputi nilai rata-rata intensitas *red, green, blue* sedangkan bentuk yaitu nilai *roundness*.

Pada fase pelatihan, ciri akan disimpan pada *database* untuk dilakukan proses pembelajaran LVQ. Pembelajaran LVQ merupakan salah satu metode pengklasifikasian pada jaringan syaraf tiruan. Dalam membuat LVQ terlebih dahulu ditentukan jumlah masukan dan keluaran pada jaringan serta beberapa parameter seperti nilai *learning rate*, nilai *maximum epoh*, dan nilai bobot.

Pada pembelajaran LVQ, data masukan berupa citra buah dengan jumlah 20 buah untuk setiap jenisnya, sehingga berjumlah 100 citra buah dengan 5 pola keluaran. Data masukan ini tentunya telah melalui tahap ekstraksi ciri. Pola keluaran mencerminkan jenis buah yang akan diidentifikasi.

Ekstraksi ciri yang didapat pada fase pengujian tidak akan tersimpan pada *database*. Hasil ekstraksi ciri ini digunakan untuk menghitung nilai *euclidean distance* terhadap nilai bobot hasil pembelajaran LVQ pada fase pelatihan. Tahap ini merupakan tahap identifikasi buah.

3.4 Pengujian dan Analisis

Pengujian dilakukan agar dapat menunjukkan bahwa aplikasi mampu bekerja sesuai dengan perancangan yang dilakukan dengan tingkat kesalahan yang sedikit. Pengujian yang dilakukan meliputi pengujian ekstraksi ciri, pengujian pelatihan LVQ, dan pengujian identifikasi buah serta analisis faktor kegagalan.

3.5 Pengambilan Kesimpulan dan Saran

Pada tahap ini, diambil kesimpulan dari hasil pengujian dan analisis terhadap Aplikasi Identifikasi Buah Berdasarkan Warna dan Bentuk Menggunakan *Webcam*. Tahap selanjutnya adalah pembuatan saran yang dimaksudkan untuk memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan aplikasi selanjutnya.

BAB IV
PERANCANGAN DAN IMPLEMENTASI

Pada tahap ini dijelaskan mengenai langkah-langkah yang akan dilakukan untuk merancang dan mengimplementasikan aplikasi identifikasi buah berdasarkan warna dan bentuk. Perancangan dan implementasi dikerjakan dengan beberapa tahap meliputi basis data, inisialisasi *webcam*, pengambilan gambar, *preprocessing*, ekstraksi ciri, pembelajaran LVQ, dan identifikasi buah. Pembuatan yang dilakukan secara bertahap bertujuan untuk memudahkan saat menganalisis setiap bagian aplikasi maupun secara keseluruhan.

4.1 Perancangan Secara Umum

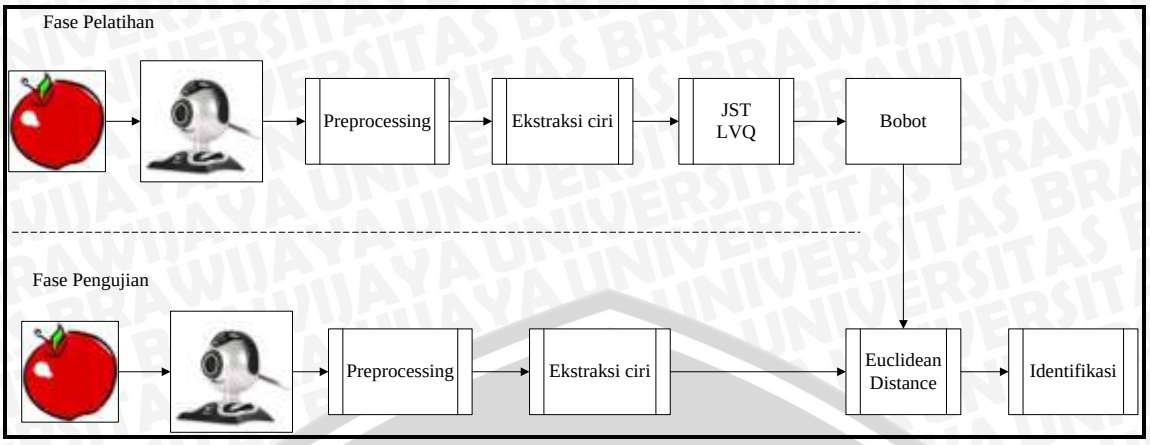
Perancangan aplikasi secara umum merupakan tahap awal dalam melakukan perancangan aplikasi identifikasi buah yang akan dibuat. Perancangan diawali dengan penggambaran blok diagram kerja sistem yang menunjukkan cara kerja aplikasi secara umum.

4.1.1 Blok diagram sistem

Pembuatan diagram sistem merupakan dasar dari perancangan sistem agar perancangan dan perealisasian aplikasi berjalan secara sistematis. Dalam sistem ini, citra masukan merupakan citra buah yang ditangkap menggunakan kamera webcam. Hal ini dapat ditunjukkan seperti pada gambar 4.1.

Gambar 4. 1 Contoh citra masukan

(**Sumber:** Perancangan)



Gambar 4. 2 Diagram proses

(**Sumber:** Perancangan)

Gambar 4.2 merupakan diagram proses dari aplikasi secara umum. Gambar 4.2 menunjukkan terdapat 2 fase yaitu fase pelatihan dan fase pengujian yang mana tidak berjalan secara bersamaan. Buah merupakan objek yang digunakan sebagai masukan sistem yang kemudian ditangkap oleh *webcam* menjadi citra buah untuk selanjutnya diproses oleh komputer. Citra buah mengalami *preprocessing* agar bisa dilakukan ekstraksi ciri dengan baik. Hasil ekstraksi ciri kemudian disimpan dalam *database* untuk dilakukan pembelajaran jaringan saraf tiruan dengan metode *Learning Vector Quantization(LVQ)* agar didapatkan bobot pada fase pelatihan. Selanjutnya membandingkan nilai bobot dengan hasil ekstraksi ciri pada fase pengujian menggunakan metode *euclidean distance* sehingga buah pada fase pengujian dapat dikenali jenisnya oleh aplikasi.

4.1.2 Cara kerja aplikasi

Cara kerja aplikasi identifikasi buah berdasarkan warna dan bentuk menggunakan *webcam* dimulai dengan proses pengambilan citra melalui *webcam*. Pengambilan citra yang dilakukan adalah meletakkan kamera di depan buah dengan posisi sejajar dan tegak lurus terhadap buah serta pada jarak yang telah ditentukan. Peletakan posisi buah saat pengambilan gambar juga telah diatur agar diperoleh hasil ekstraksi ciri yang sesuai.

Pengambilan citra melalui *webcam* berdasarkan *Direct Show* yang dibuat oleh Andrew Kirilov yang mana harus dilakukan inisialisasi *webcam* terlebih dahulu. Komponen *Direct Show* dapat berfungsi untuk merubah citra video menjadi citra *bitmap*

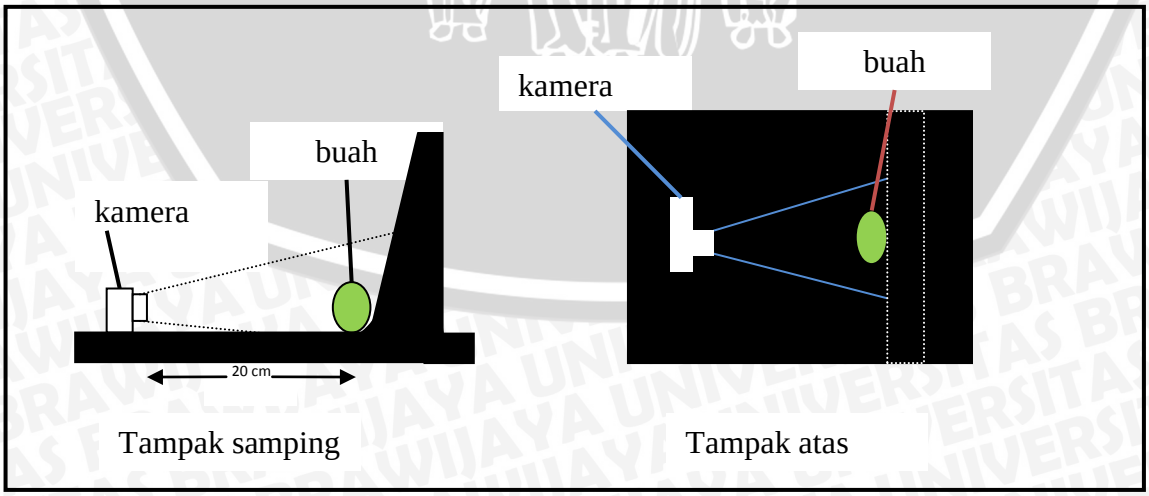
dengan *capture image* agar dapat diproses dengan teknik pengolahan citra saat citra buah dipisahkan dengan bagian *background*.

Setelah buah terpisahkan dengan latar belakangnya, maka dilakukan ekstraksi ciri warna dan bentuk. Ciri warna yang diambil adalah nilai rata-rata intensitas pixel kanal RGB sedangkan ciri bentuk adalah nilai *roundness* yang didapat berdasarkan nilai luas pixel objek dan keliling pixel objek. Hasil ekstraksi ciri akan disimpan dalam *database* untuk proses pelatihan LVQ.

Dalam proses pelatihan LVQ dibutuhkan suatu masukan vektor yaitu vektor ciri hasil proses ektraksi. Setiap masukan memiliki vektor ciri yang direpresentasikan dalam matriks berukuran 1x4, sehingga akan terdapat matrik berukuran 100x4 saat pelatihan. Selanjutnya *user* dapat mengatur nilai parameter *learning rate*, dan *maximum epoch*. Jika LVQ sudah dianggap siap oleh user, maka bobot bisa disimpan untuk proses identifikasi. Proses identifikasi dilakukan dengan menghitung jarak *euclidean* antara bobot setiap jenis buah dengan buah uji. Jika didapatkan jarak terdekat terhadap salah satu jenis buah maka buah uji akan masuk ke dalam jenis buah tersebut.

4.1.3 Peletakan Objek

Proses pengolahan citra dan ekstraksi ciri dapat bekerja secara maksimal jika citra memiliki karakteristik tertentu. Citra yang diolah harus berupa citra dua dimensi dengan perspektif tegak lurus. Untuk mendapatkan citra yang sesuai dengan keinginan, diperlukan peletakan *webcam* dan objek yang tepat pada tempat sistem diimplementasikan. Buah akan diletakkan pada suatu *light tent* dengan latar belakang berwarna putih. Gambaran umum konfigurasi seperti Gambar 4.3.



Gambar 4. 3 Posisi peletakkan kamera dan buah

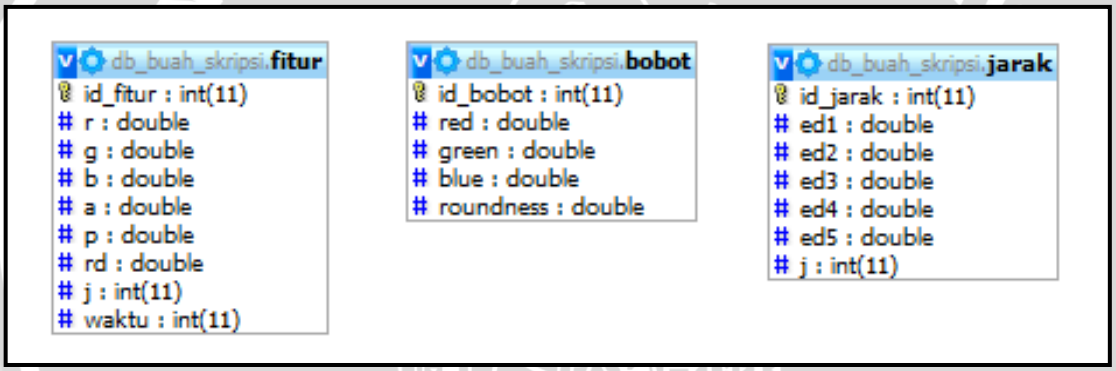
(**Sumber:** Perancangan)

4.2 Perancangan Perangkat Lunak

Pada sub bab perancangan perangkat lunak akan membahas secara detail tentang setiap tahapan proses pada aplikasi identifikasi buah berdasarkan warna dan bentuk. Beberapa tahap perancangan tersebut meliputi basis data, inisialisasi *webcam* dan pengambilan gambar, pemisahan *background*, ekstraksi ciri, pembelajaran *Learning Vector Quantization(LVQ)*, identifikasi buah. Perancangan perangkat lunak diimplementasikan dalam bahasa pemrograman C#.

4.2.1 Basis data

Perancangan basis data dibangun menggunakan MySQL. Basis data ini akan digunakan sebagai tempat penyimpanan hasil ekstraksi ciri dan juga bobot. Beberapa tabel yang dibutuhkan adalah tabel fitur, bobot, dan jarak seperti yang ditunjukkan pada gambar 4.4.



Gambar 4. 4 Tabel basis data

(**Sumber:** Perancangan)

Gambar 4.4 menunjukkan bahwa tabel fitur akan menyimpan nilai hasil ekstraksi ciri seperti *red*(r), *green*(g), *blue*(b), *area*(a), *perimeter*(p), *roundness*(rd) yang bertipe data double. Selain itu tabel fitur juga akan menyimpan beberapa informasi penting seperti jenis buah(j) dan waktu pemrosesan ekstraksi ciri(waktu). Tabel bobot menyimpan bobot hasil pelatihan LVQ untuk setiap jenis buah yaitu nilai *red*, *green*, *blue*, dan *roundness*. Tabel jarak menyimpan nilai jarak setiap pelatihan LVQ dimana terdapat jarak terhadap kelas 1 sampai kelas 5 yang diwakili dengan ed1, ed2, ed3, ed4, dan ed5. Ketiga tabel ini akan diimplementasikan menggunakan *Data Definition Lenguage*(DDL) MySQL. DDL untuk membuat suatu basis data adalah sebagai berikut:

```
CREATE DATABASE db_buah_skripsi;
```

DDL untuk membuat tabel fitur adalah sebagai berikut:

```
CREATE TABLE fitur
(
 id_fitur int(11) NOT NULL AUTO_INCREMENT,
 r double NOT NULL,
 g double NOT NULL,
 b double NOT NULL,
 a double NOT NULL,
 p double NOT NULL,
 rd double NOT NULL,
 j int(11) NOT NULL,
 waktu int(11) NOT NULL,
 PRIMARY KEY (id_fitur)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1;
```

DDL untuk membuat tabel bobot adalah sebagai berikut:

```
CREATE TABLE bobot
(
 id_bobot int(11) NOT NULL AUTO_INCREMENT,
 red double NOT NULL,
 green double NOT NULL,
 blue double NOT NULL,
 roundness double NOT NULL,
 PRIMARY KEY (id_bobot)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1;
```

DDL untuk membuat tabel jarak adalah sebagai berikut:

```
CREATE TABLE jarak
(
 id_jarak int(11) NOT NULL AUTO_INCREMENT,
 ed1 double NOT NULL,
 ed2 double NOT NULL,
 ed3 double NOT NULL,
 ed4 double NOT NULL,
 ed5 double NOT NULL,
 PRIMARY KEY (id_jarak)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1;
```

Menghubungkan basis data dengan perangkat lunak pada bahasa C# menggunakan MySQL *Connector*. Berikut ini adalah bagian kode bahasa C# agar dapat terhubung dengan basis data:

```
//inisialisasidatabase
string connect_db =
"server=localhost;user=root;database=db_buah_skripsi;port=3306;password='';";
MySqlConnection con = new MySqlConnection(connect_db);
```

4.2.2 Inisialisasi webcam dan capture image

Perancangan inisialisasi *webcam* dilakukan dengan memanfaatkan beberapa fitur komponen *Direct Show* yang dibuat oleh Andrew Kirillov (*developer* Aforge *framework*) sebagai antar muka perangkat keras. *Direct Show* adalah pustaka

Component Object Model (COM) untuk dapat mengakses perangkat keras berbasis *Windows Driver Model* (WDM) seperti *webcam*. Berikut ini adalah pengelompokan class yang ada di dalam *Direct Show* dapat dilihat pada Tabel 4.1.

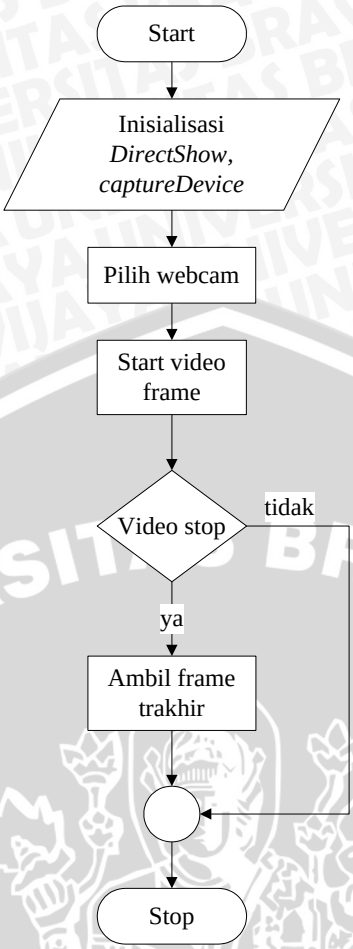
Tabel 4.1 Daftar class interface yang ada di dalam *Direct Show*

No	Class	Keterangan
1	FileVideoSource	Sumber video untuk file video
2	FilterCategory	Kategori *Direct Show* filter
3	FilterInfo	Informasi Filter *Direct Show*
4	FilterInfoColection	Koleksi objek informasi filter
5	VideoCapabilities	Kemampuan perangkat video seperti ukuran frame dan frame rate
6	Video*Capture*Device	Sumber video untuk pengambilan gambar video pada perangkat local (*webcam*)

(**Sumber:** AForge .NET Framework)

Ada beberapa filter yang digunakan dalam menangkap gambar buah menggunakan *webcam* yang membentuk suatu graf filter yaitu filter sumber, filter pengubah dan filter hasil. Filter sumber untuk mengakses perangkat *webcam*. Filter pengubah melakukan kompresi data gambar raw menjadi RGB (Red Green Blue). Filter hasil berupa tempat untuk meletakkan data gambar.

Hasil tangkapan yang diletakkan pada obyek berupa aliran data *real–time* pada memori. Agar dapat diolah maka harus diubah menjadi file gambar. Data dalam obyek pada satu waktu dicuplik ke dalam sebuah array kosong dan diberi format gambar *bitmap* (BMP). Flowchart inisialisasi *webcam* seperti yang ditunjukkan pada Gambar 4.5.



Gambar 4. 5 Flowchart inisialisasi *webcam* dan *capture image*

(**Sumber:** Perancangan)

Implementasi inisialisasi *webcam* dan pengambilan gambar pada bahasa C# adalah sebagai berikut:

```
using AForge.Video.DirectShow;
```

```
VideoCaptureDevices = new FilterInfoCollection(FilterCategory.VideoInputDevice);
```

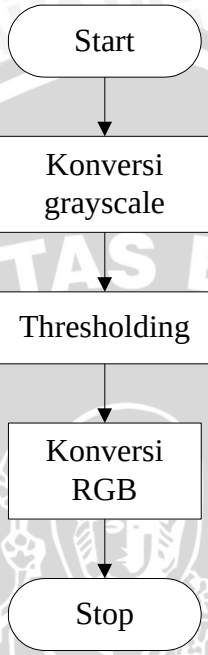
Inisialisasi *webcam* yang memanfaatkan fitur *DirectShow* terlebih dahulu memanggil library *DirectShow* dari AForge. Kemudian membuat suatu objek berupa filter yang mengakses perangkat *webcam*.

```
FinalVideo.Start();
FinalVideo.Stop();
```

Untuk menjalankan video dapat memanggil fungsi *Start()* dan untuk mengehentikan video dengan cara memanggil fungsi *Stop()*. Dengan adanya pemanggilan fungsi *Stop()* maka akan diambil frame terakhir dari video dan frame inilah yang akan diolah pada proses selanjutnya.

4.2.3 Preprocessing

Output dari tahap *capture image* selanjutnya akan menjadi input dari tahap *preprocessing*. *Preprocessing* yang dilakukan bertujuan memisahkan citra buah dengan latar belakangnya. Proses *preprocessing* yang berlangsung yaitu konversi *grayscale*, *thresholding* dan konversi RGB seperti pada gambar 4.6.

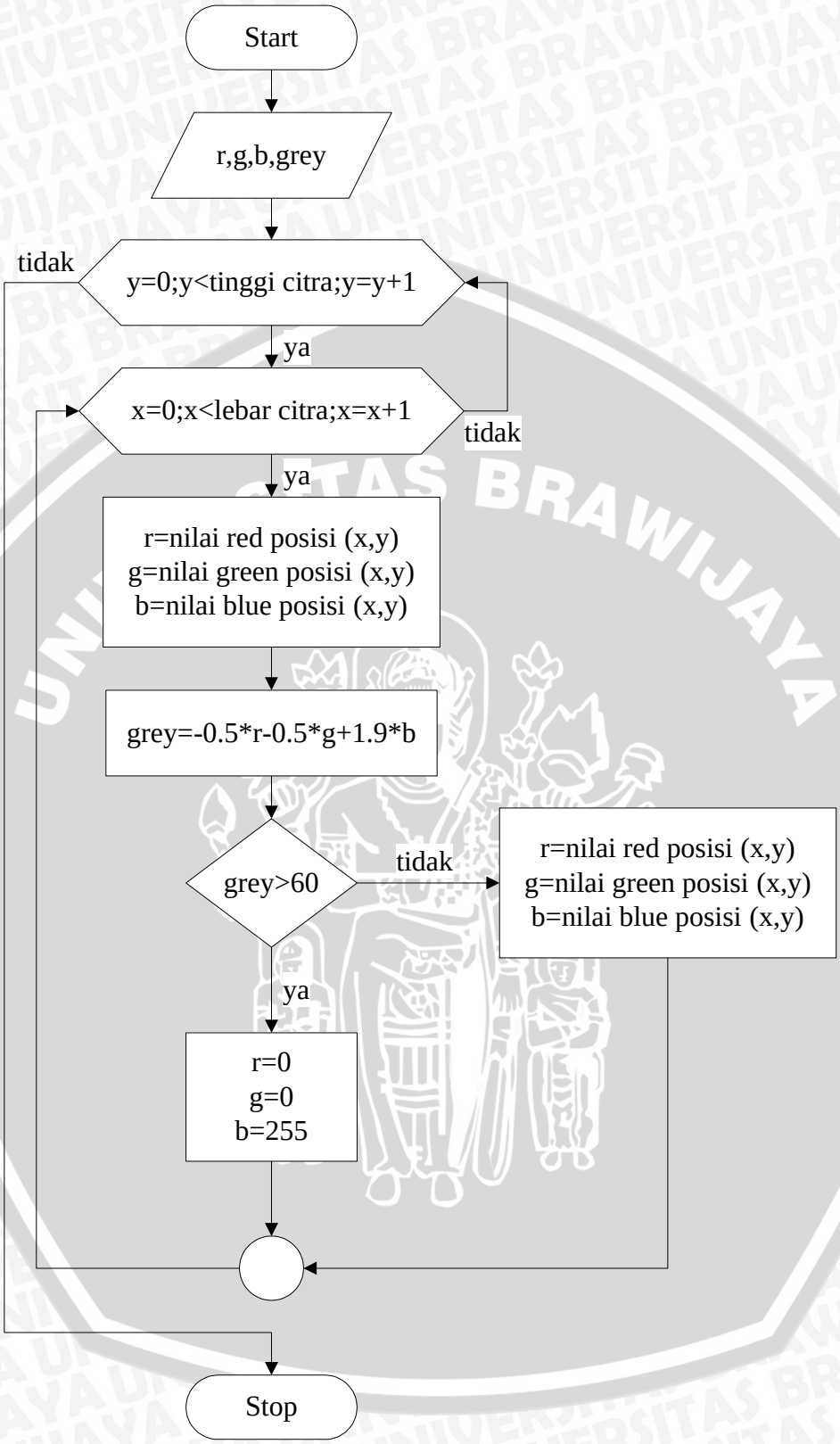


Gambar 4. 6 Flowchart *preprocessing*

(**Sumber:** perancangan)

Gambar 4.7 menjelaskan bahwa citra yang menjadi masukan akan dilakukan konversi ke dalam bentuk *grayscale* karena hasil pada tahap *capture image* berupa citra warna atau RGB. Tujuan konversi *grayscale* ini adalah supaya kombinasi warna menjadi lebih sederhana sehingga memudahkan untuk proses selanjutnya. Perhitungan *grayscale* yaitu$-0.5 * R - 0.5 * G + 1.9 * B$. *Grayscale* menyebabkan citra warna menjadi citra abu-abu yang memiliki rentang nilai piksel mulai 0 – 255.

Selanjutnya adalah melakukan *thresholding* atau pengambangan. Proses *thresholding* bertujuan untuk menghilangkan latar belakang citra. Nilai *threshold* ideal pada proses ini adalah 60, yang didapatkan dengan melakukan *trial* dan *error*. Perhitungan *threshold* adalah dengan mengubah piksel bernilai >60 menjadi berwarna biru (0, 0, 255). Sedangkan piksel bernilai ≤60 akan dikembalikan menjadi warna aslinya.



Gambar 4.7 Flowchart pemisahan *background*

(**Sumber:** Perancangan)

Implementasi proses *preprocessing* pada bahasa C# adalah sebagai berikut:

```
gambar_pisah = (Bitmap)pictureBox2.Image.Clone();

for (int y = 0; y < gambar_pisah.Height; y++)
{
 for (int x = 0; x < gambar_pisah.Width; x++)
 {
 //mengambil nilai RGB pixel
 int r = gambar_pisah.GetPixel(x, y).R;
 int g = gambar_pisah.GetPixel(x, y).G;
 int b = gambar_pisah.GetPixel(x, y).B;

 //grayscale
 int abu = -0.5*r – 0.5 * g + 1.9 * b;

 //threshold
 if (abu > 60)
 {
 r = 0;
 g = 0;
 b = 255;
 }

 //RGB
 gambar_pisah.SetPixel(x, y, Color.FromArgb(r, g, b));
 }
}
pictureBox2.Image = gambar_pisah;
```

4.2.4 Perancangan ekstraksi ciri

Pada tahap ekstraksi ciri, citra masukan berupa citra hasil pemisahan *background*. Ekstraksi ciri bertujuan untuk mendapatkan karakteristik dari suatu buah yang dapat membedakan dengan buah yang lain. Ciri yang digunakan adalah ciri warna dan ciri bentuk. Ciri warna yang diambil adalah *mean* warna RGB, sedangkan ciri bentuk adalah nilai *roundness*. Untuk mendapatkan ciri warna dan bentuk digunakan beberapa metode seperti yang dijelaskan pada gambar 4.8.

Gambar 4.8 Flowchart ekstraksi ciri

(**Sumber:** Perancangan)

Gambar 4.8 menunjukkan bahwa citra hasil pemisahan *backgound* merupakan masukan dalam tahap ekstraksi ciri. Setelah mendapat objek yang diinginkan maka dilakukan perhitungan mean warna. Selanjutnya citra dikonversi ke dalam bentuk biner untuk memudahkan dalam proses selanjutnya. Apabila perhitungan *area* selesai maka dilakukan operasi morfologi terhadap citra untuk memperoleh *perimeter* sehingga akan diperoleh nilai *roundness*.

4.2.4.1 Ciri warna

Ekstraksi ciri warna didapat dengan menghitung intensitas rata-rata pixel kanal merah, hijau dan biru pada buah dengan rumus sebagai berikut:

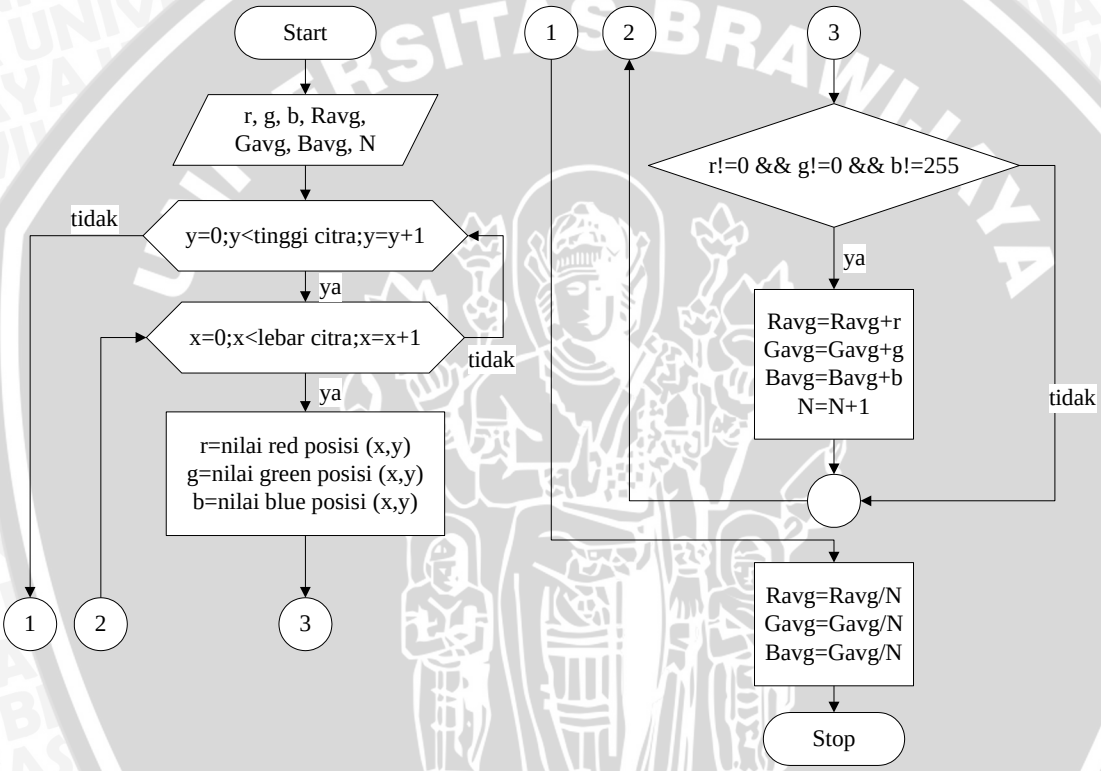
$$r_{avg} = \frac{\sum_{i=1}^{N} r(i)}{N}; g_{avg} = \frac{\sum_{i=1}^{N} g(i)}{N}; b_{avg} = \frac{\sum_{i=1}^{N} b(i)}{N}$$

dengan:

r_{avg}, g_{avg}, b_{avg} = rata-rata intensitas warna R, G, B

r(i), g(i), b(i) = nilai intensitas warna R, G, B pada suatu pixel

N = total pixel pada objek.



Gambar 4.9 Flowchart ekstraksi ciri warna

(**Sumber:** Perancangan)

Gambar 4.9 menjelaskan bahwa ekstraksi ciri warna dilakukan dengan cara mengambil setiap nilai RGB pixel pada objek. Pengambilan nilai dengan cara memeriksa pixel yang tidak berwarna biru atau bernilai R=0, G=0, B=255. Selanjutnya nilai RGB disimpan dalam suatu variabel yang nantinya akan dijumlahkan dengan nilai RGB pada iterasi selanjutnya. Proses ini berlangsung selama belum mencapai pada koordinat pixel terakhir. Setelah perulangan berakhir maka dilakukan perhitungan nilai rata-rata RGB yang didapat dengan cara membagi jumlah nilai RGB dari perulangan

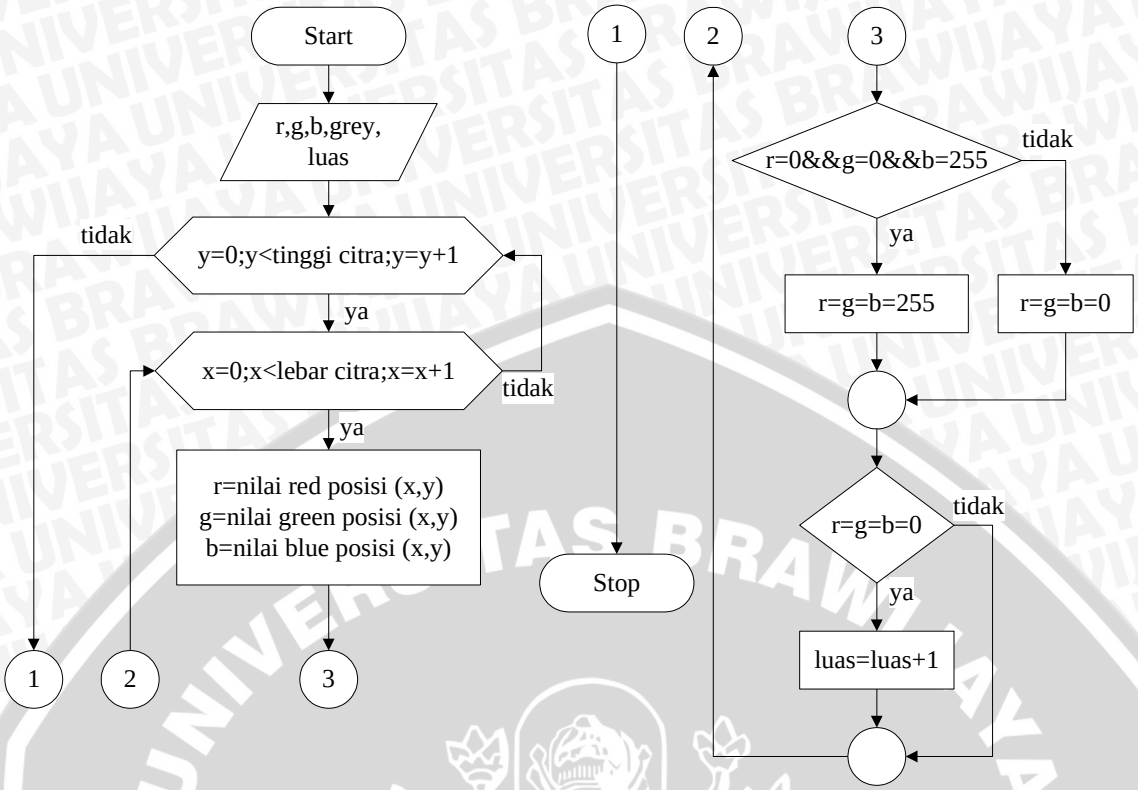
dengan jumlah pixel buah pada citra. Implementasi proses ekstraksi ciri warna pada bahasa C# adalah sebagai berikut:

```
gambar_ekstraksi = (Bitmap)pictureBox2.Image.Clone();
jumlah = 0;

for (int y = 0; y < gambar_ekstraksi.Height; y++)
{
 for (int x = 0; x < gambar_ekstraksi.Width; x++)
 {
  //mengambil nilai RGB pixel
  red = gambar_ekstraksi.GetPixel(x, y).R;
  green = gambar_ekstraksi.GetPixel(x, y).G;
  blue = gambar_ekstraksi.GetPixel(x, y).B;

  //memeriksa kondisi nilai RGB pixel
  if (red != 0 && green != 0 && blue != 255)
  {
   Ravg += red;
   Gavg += green;
   Bavg += blue;
   jumlah = jumlah + 1;
  }
 }
}

//menghitung nilai rata-rata RGB pixel objek buah
Ravg = Ravg / jumlah;
Gavg = Gavg / jumlah;
Bavg = Bavg / jumlah;
```

4.2.4.2 Ciri bentuk

Ciri bentuk yang diambil dari buah adalah nilai kebundaran (*roundness*). Nilai kebundaran didapat dengan mencari terlebih dahulu *area* dan *perimeter* buah. Ekstraksi ciri bentuk dimulai dengan merubah citra warna ke *grayscale* kemudian merubahnya lagi ke dalam bentuk biner. Selanjutnya menghitung *area* dan *perimeter* buah.

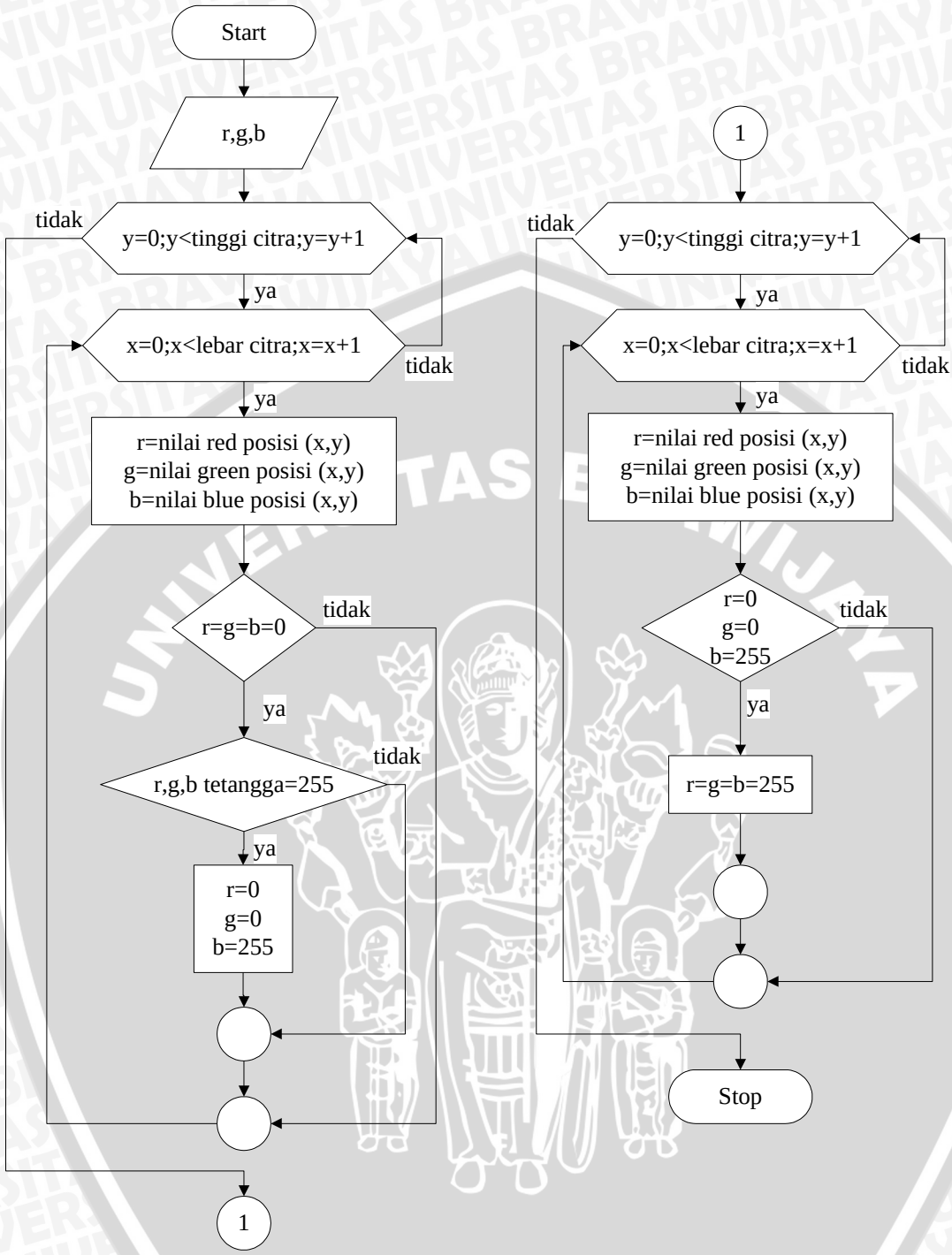
Nilai *perimeter* dan *area* diestimasikan dalam pixel. *Perimeter* didapat dengan cara menghitung jumlah pixel yang berbatasan dengan *background*, sedangkan *area* didapat dengan cara menghitung jumlah pixel yang tertutup oleh keliling buah. Untuk mendapatkan nilai *perimeter* digunakan metode operasi morfologi *boundary detection*, dimana melakukan erosi terhadap objek buah kemudian mengurangi citra sebelum erosi dengan citra hasil erosi.



Gambar 4.10 Flowchart menghitung luas buah

(**Sumber:** Perancangan)

Gambar 4.10 merupakan proses ekstraksi bentuk untuk mendapatkan luas (*area*) buah. Luas buah merupakan jumlah pixel yang menyatakan buah sehingga perlu dilakukan pengecekan setiap pixel pada buah. Untuk memudahkan pengecekan maka terlebih dahulu buah dikonversi ke dalam bentuk biner dengan cara mengubah setiap pixel berwarna biru (R=0, G=0, B=255) menjadi berwarna putih(R=G=B=255) dan selain itu menjadi berwarna hitam(R=G=B=0). Jumlah nilai pixel berwarna hitam inilah yang merupakan luas dari objek buah.



Gambar 4.11 Flowchart operasi morfologi erosi

(**Sumber:** perancangan)

Gambar 4.11 merupakan salah satu tahapan proses ekstraksi bentuk untuk mendapatkan keliling (*perimeter*) buah yaitu operasi morfologi erosi. Tujuan dari erosi adalah mengurangi 1 pixel terluar dari buah. Proses ini dimulai dari mendapatkan nilai biner setiap pixel pada citra kemudian melakukan pemeriksaan untuk setiap nilai biner

pada buah yaitu 1. Jika ada pixel tetangga yang bernilai 0 maka keluaran pixel diset menjadi 0.

Setelah didapat citra hasil erosi, selanjutnya adalah mengurangi citra sebelum erosi dengan citra hasil erosi sehingga akan didapatkan pixel keliling dari buah seperti yang ditunjukkan pada gambar 4.12. Proses ini dinamakan dengan operasi morfologi *boundary detection*. Proses selanjutnya yaitu mendapatkan nilai keliling dengan cara menghitung jumlah pixel buah hasil *boundary detection* sehingga nilai *roundness* buah dapat dicari menggunakan rumus:

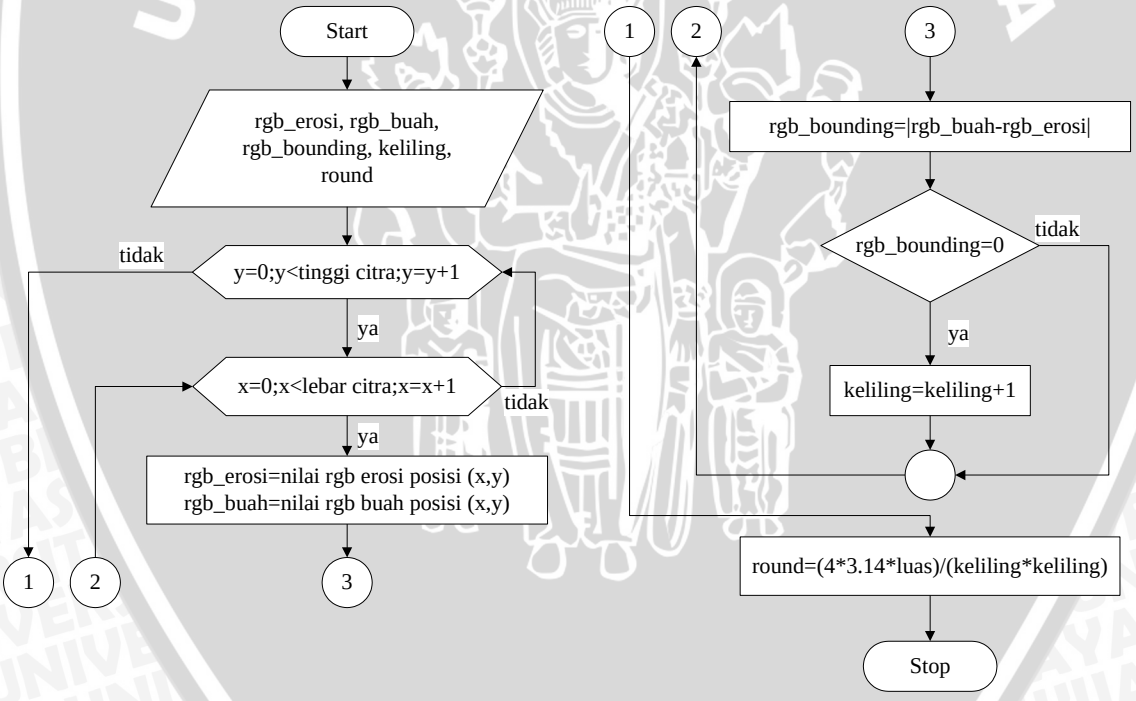
$$C = 4\pi \frac{Area}{Perimeter^2}$$

dengan:

C = kebundaran buah

Area = luas buah

Perimeter = keliling buah



Gambar 4.12 Flowchart menghitung keliling dan *roundness* buah

(**Sumber:** perancangan)

Implementasi proses ekstraksi ciri bentuk pada bahasa C# adalah sebagai berikut:

```
gambar_ekstraksi = (Bitmap)pictureBox2.Image.Clone();
l = 0;
for (int y = 0; y < gambar_ekstraksi.Height; y++)
```

```
{
 for (int x = 0; x < gambar_ekstraksi.Width; x++)
 {
  //mengambil nilai RGB pixel
  int r = gambar_ekstraksi.GetPixel(x, y).R;
  int g = gambar_ekstraksi.GetPixel(x, y).G;
  int b = gambar_ekstraksi.GetPixel(x, y).B;

  //memeriksa kondisi nilai RGB pixel
  if (r == 0 && g == 0 && b == 255)
  {
   l = l + 1;
   r = 255;
   g = 255;
   b = 255;
  }
  Else
  {
   l = l + 1;
   r = 0;
   g = 0;
   b = 0;
  }
  gambar_ekstraksi.SetPixel(x, y, Color.FromArgb(r, g, b));
 }
}
pictureBox3.Image = gambar_ekstraksi;

gambar_erosi = (Bitmap)pictureBox3.Image.Clone();

for (int y = 0; y < gambar_erosi.Height; y++)
{
 for (int x = 0; x < gambar_erosi.Width; x++)
 {
  int r = gambar_erosi.GetPixel(x, y).R;
  int g = gambar_erosi.GetPixel(x, y).G;
  int b = gambar_erosi.GetPixel(x, y).B;

  if (r == 0 && g == 0 && b == 0)
  {
   if (x - 1 != -1 && y - 1 != -1 && x + 1 < gambar_erosi.Width && y + 1 <
gambar_erosi.Height)
   {
    if (y + 1 < gambar_erosi.Height || x + 1 < gambar_erosi.Width || y - 1 <
gambar_erosi.Height || x - 1 < gambar_erosi.Width)
    {
     //kiri atas
     int rkia = gambar_erosi.GetPixel(x - 1, y - 1).R;
     int gkia = gambar_erosi.GetPixel(x - 1, y - 1).G;
     int bkia = gambar_erosi.GetPixel(x - 1, y - 1).B;

     //atas
     int ra = gambar_erosi.GetPixel(x, y - 1).R;
     int ga = gambar_erosi.GetPixel(x, y - 1).G;
     int ba = gambar_erosi.GetPixel(x, y - 1).B;

     //kanan atas
     int rkaa = gambar_erosi.GetPixel(x + 1, y - 1).R;
     int gkaa = gambar_erosi.GetPixel(x + 1, y - 1).G;
     int bkaa = gambar_erosi.GetPixel(x + 1, y - 1).B;

     //kiri
     int rki = gambar_erosi.GetPixel(x - 1, y).R;
     int gki = gambar_erosi.GetPixel(x - 1, y).G;
     int bki = gambar_erosi.GetPixel(x - 1, y).B;
```

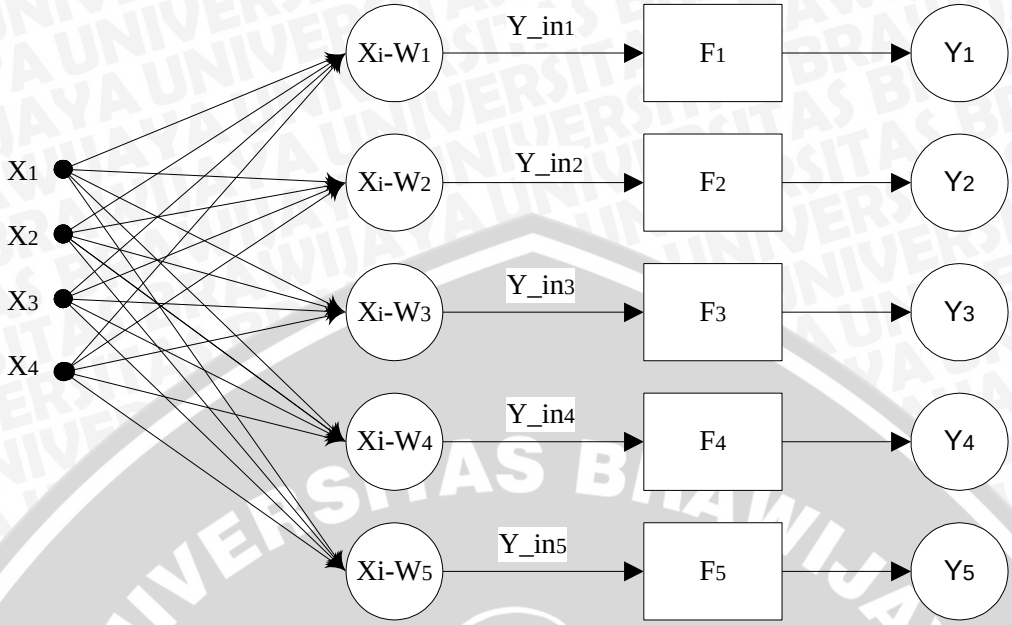
```
        //kanan
        int rka = gambar_erosi.GetPixel(x + 1, y).R;
        int gka = gambar_erosi.GetPixel(x + 1, y).G;
        int bka = gambar_erosi.GetPixel(x + 1, y).B;


        //kiri bawah
        int rkib = gambar_erosi.GetPixel(x - 1, y + 1).R;
        int gkib = gambar_erosi.GetPixel(x - 1, y + 1).G;
        int bkib = gambar_erosi.GetPixel(x - 1, y + 1).B;

        //bawah
        int rb = gambar_erosi.GetPixel(x, y + 1).R;
        int gb = gambar_erosi.GetPixel(x, y + 1).G;
        int bb = gambar_erosi.GetPixel(x, y + 1).B;

        //kanan bawah
        int rkab = gambar_erosi.GetPixel(x + 1, y + 1).R;
        int gkab = gambar_erosi.GetPixel(x + 1, y + 1).G;
        int bkab = gambar_erosi.GetPixel(x + 1, y + 1).B;

        if ((ra == 255 && ga == 255 && ba == 255)||(rki == 255 && gki == 255 && bki ==
255)||(rka == 255 && gka == 255 && bka == 255)||(rb == 255 && gb == 255 && bb == 255)
|| (rkia == 255 && gkia == 255 && bkia == 255) || (rkaa == 255 && gkaa == 255 && bkaa
== 255) || (rkib == 255 && gkib == 255 && bkib == 255) || (rkab == 255 && gkab == 255
&& bkab == 255))
          {
           r = 0;
           g = 0;
           b = 255;
          }
         }
        }
       }
       gambar_erosi.SetPixel(x, y, Color.FromArgb(r, g, b));
      }
     }

     for (int y = 0; y < gambar_erosi.Height; y++)
     {
      for (int x = 0; x < gambar_erosi.Width; x++)
      {
       int r = gambar_erosi.GetPixel(x, y).R;
       int g = gambar_erosi.GetPixel(x, y).G;
       int b = gambar_erosi.GetPixel(x, y).B;

       if (r == 0 && g == 0 && b == 255)
       {
        r = 255;
        g = 255;
        b = 255;
       }
       gambar_erosi.SetPixel(x, y, Color.FromArgb(r, g, b));
      }
     }
     pictureBox4.Image = gambar_erosi;

     gambar_erosi = (Bitmap)pictureBox4.Image.Clone();
     gambar_ekstraksi = (Bitmap)pictureBox3.Image.Clone();

     for (int y = 0; y < gambar_ekstraksi.Height; y++)
     {
      for (int x = 0; x < gambar_ekstraksi.Width; x++)
      {
       int r = gambar_ekstraksi.GetPixel(x, y).R;
       int g = gambar_ekstraksi.GetPixel(x, y).G;
```

```
  int b = gambar_ekstraksi.GetPixel(x, y).B;

  int re = gambar_erosi.GetPixel(x, y).R;
  int ge = gambar_erosi.GetPixel(x, y).G;
  int be = gambar_erosi.GetPixel(x, y).B;

  Rhasil = Math.Abs(r - re);
  Ghasil = Math.Abs(g - ge);
  Bhasil = Math.Abs(b - be);

  gambar_ekstraksi.SetPixel(x, y, Color.FromArgb(Rhasil, Ghasil, Bhasil));
 }
}
pictureBox4.Image= gambar_ekstraksi;

for (int y = 0; y < gambar_ekstraksi.Height; y++)
{
 for (int x = 0; x < gambar_ekstraksi.Width; x++)
 {
  red = gambar_ekstraksi.GetPixel(x, y).R;
  green = gambar_ekstraksi.GetPixel(x, y).G;
  blue = gambar_ekstraksi.GetPixel(x, y).B;

  if (red == 255 && green == 255 && blue == 255)
  {
   k = k + 1;
  }
 }
}
roundness = (double)((4 * Math.PI * l)/(k*k));
```

Sebanyak 100 buah pelatihan dan 23 buah pengujian mengalami ekstraksi ciri secara bergantian. Buah pelatihan mengalami ekstraksi ciri dengan urutan acak. 100 buah pelatihan hasil ekstraksi ciri ini akan disimpan nilai-nilainya ke dalam basis data.

4.2.5 Perancangan pembelajaran *Learning Vector Quantization*

Proses pembelajaran LVQ yang dilakukan pertama kali adalah pengambilan data training. Data training ini diambil dari basis data yang berisi nilai-nilai ekstraksi ciri. Data training ini merupakan input *layer* pada jaringan saraf tiruan seperti yang ditunjukkan pada gambar 4.13.



Gambar 4.13 Struktur jaringan LVQ
(**Sumber:** Perancangan)

Gambar 4.13 adalah arsitektur jaringan LVQ (*Learning Vector Quantization*) pada aplikasi yang terdiri dari 4 *node* lapisan masukan (*input layer*), 5 *node* lapisan tersembunyi (*hidden layer*) serta 5 *node* lapisan keluaran (*output layer*). Lapisan input memiliki 4 node yang disimbolkan dengan nilai x_1, x_2, x_3, dan x_4. Pada tiap lapisan masukan terlebih dahulu diberikan 5 nilai bobot yang berbeda yaitu W_1, W_2, W_3, W_4, dan W_5. Pemberian bobot bertujuan agar nilai tiap masukan memiliki penimbang (bobot) yang kemudian akan dihitung nilainya dan diselesaikan dengan persamaan yang dimiliki oleh metode LVQ (*Learning Vector Quantization*), sehingga jaringan memiliki 5 buah kelas yang berbeda, yaitu kelas Y_1, Y_2, Y_3, Y_4, Y_5. Kelas Y_1 memiliki bobot W_1, kelas Y_2 memiliki bobot W_2 dan seterusnya. Keluaran dari lapisan ini akan menjadi masukan bagi lapisan tersembunyi (*hidden layer*) sebanyak 5 *node* yaitu Y_in_1, Y_in_2, Y_in_3, Y_in_4 dan Y_in_5.

Bobot awal pada aplikasi ini diberikan dengan mengambil nilai data pelatihan pertama untuk setiap kelasnya. Dengan menggunakan metode *euclidean distance* bahwa nilai paling kecil yang dihasilkan adalah pemenang dan merupakan kelas dari input tersebut maka pada lapisan keluaran (*output layer*) digunakan sebuah fungsi pembanding yang berguna membandingkan dua nilai tersebut untuk dicari nilai terkecilnya. Dalam gambar 4.13 fungsi pembanding tersebut dituliskan dengan simbol

F_1, F_2, F_3, F_4, dan F_5.Pada aplikasi ini nilai *learning rate* ditentukan oleh user begitu juga nilai pengurangan *learning rate*.

$$d_{ij} = \sqrt{\left(x_{iR} - w_{jR}\right)^2 + \left(x_{iG} - w_{jG}\right)^2 + \left(x_{iB} - w_{jB}\right)^2 + \left(x_{iround} - w_{jround}\right)^2}$$

dengan:

d_{ij} = euclidean distance input(X_i) dengan bobot(W_j)

x_{iR} = nilai vektor input red ke-i

x_{iG} = nilai vektor input green ke-i

x_{iB} = nilai vektor input blue ke-i

x_{iR} = nilai vektor input round ke-i

w_{iR} = nilai vektor bobot red ke-j

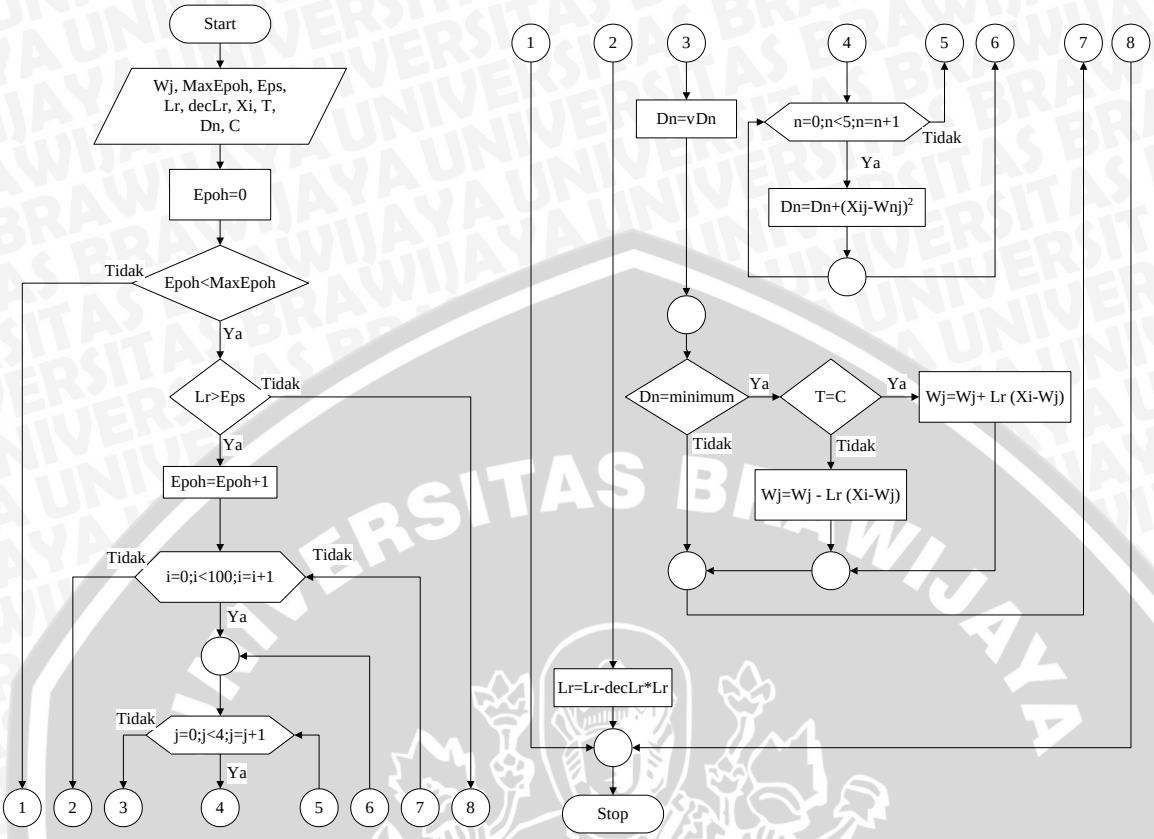
w_{iG} = nilai vektor bobot green ke-j

w_{iB} = nilai vektor bobot blue ke-j

w_{iR} = nilai vektor bobot round ke-j

Nilai input yang diberikan ke lapisan masukan (input layer) berupa vektor. Jumlah dari vektor x sebagai input tergantung jumlah input yang diberikan kelapisan masukan yang berasal dari matriks ukuran 100 x 4 dan diubah menjadi berukuran 1x4. Tujuannya adalah untuk mempermudah penentuan jumlah node pada lapisan masukan serta perhitungannya dengan metode LVQ (Learning Vector Quantization).

4.2.5.1 Algoritma LVQ



Gambar 4.14 Flowchart algoritma LVQ

(**Sumber:** Perancangan)

Gambar 4.14 menjelaskan algoritma jaringan saraf tiruan metode LVQ. Langkah pertama adalah menentukan masing-masing kelas output (C), bobot (W), *learning rate* (Lr), maksimum epoh (MaxEpoh), dan eror minimum yang diharapkan (Eps). Kemudian memasukkan data input (X_i) dengan i=1, 2, 3, ..., 100 serta target kelas (T).

Masing-masing input dibandingkan dengan bobot yang telah ditetapkan dengan melakukan pengukuran jarak antara masing-masing bobot W dan input X. Nilai minimum dari hasil perbandingan akan menentukan kelas dari vektor input dan perubahan bobot dari kelas tersebut. Untuk input dan bobot yang memiliki kelas yang sama maka bobot diubah menjadi $W = W + Lr(X - W)$ sedangkan untuk input dan bobot dengan kelas yang berbeda diubah menjadi $W = W - Lr(X - W)$. Proses ini dilakukan secara terus-menerus sampai kondisi maxepoh dan learning rate terpenuhi.

Berikut ini adalah salah satu contoh pengukuran dan pembandingan jarak pada aplikasi identifikasi buah berdasarkan warna dan bentuk menggunakan webcam untuk 3 input pertama.

Vektor input buah:

- Apel manalagi = [202.42 171.36 89.69 0.44]
- Apel manalagi = [156.39 133.85 40.39 0.58]
- Pisang susu = [184.67 122.52 25.94 0.35]

Vektor bobot masing-masing jenis buah:

- Apel manalagi = [181.21 166.99 80.10 0.51]
- Mangga manalagi= [125.95 119.05 51.30 0.49]
- Jambu merah = [190.65 133.00 48.83 0.54]
- Jeruk manis = [226.10 131.45 25.08 0.59]
- Pisang susu = [201.25 132.55 38.66 0.32]

Proses pelatihan yang terjadi adalah sebagai berikut:

Iterasi 1:

Data ke-1 [202.42 171.36 89.69 0.44] mewakili buah apel manalagi:

- Jarak ke bobot 1 (jenis buah apel manalagi)

$$= \sqrt{(202.42 - 181.21)^2 + (171.36 - 166.99)^2 + (89.69 - 80.10)^2 + (0.44 - 0.51)^2}$$

$$= 23.68405$$

- Jarak ke bobot 2 (jenis buah mangga manalagi)

$$= \sqrt{(202.42 - 125.95)^2 + (171.36 - 119.05)^2 + (89.69 - 51.30)^2 + (0.44 - 0.49)^2}$$

$$= 100.2886$$

- Jarak ke bobot 3 (jenis buah jambu merah)

$$= \sqrt{(202.42 - 190.65)^2 + (171.36 - 131.45)^2 + (89.69 - 48.83)^2 + (0.44 - 0.54)^2}$$

$$= 57.26755$$

- Jarak ke bobot 4 (jenis buah jeruk manis)

$$= \sqrt{(202.42 - 226.10)^2 + (171.36 - 166.99)^2 + (89.69 - 25.08)^2 + (0.44 - 0.59)^2}$$

$$= 79.54888$$

- Jarak ke bobot 5 (jenis buah pisang susu)

$$= \sqrt{(202.42 - 201.25)^2 + (171.36 - 132.55)^2 + (89.69 - 38.66)^2 + (0.44 - 0.32)^2}$$

$$= 64.12223$$

Jarak terpendek adalah pada bobot ke-1 dengan target kelas adalah 1, sehingga bobot ke-1 yang baru adalah:

$w_{11} = w_{11} - Lr * (x_{11} - w_{11}) = 181.21 - 0.9 * (181.21 - 202.42) = 162.12$

$w_{12} = w_{12} - Lr * (x_{12} - w_{12}) = 166.99 - 0.9 * (166.99 - 171.36) = 163.06$

$w_{13} = w_{13} - Lr * (x_{13} - w_{13}) = 80.10 - 0.9 * (80.10 - 89.69) = 71.65$

$w_{14} = w_{14} - Lr * (x_{14} - w_{14}) = 0.51 - 0.9 * (0.51 - 0.44) = 0.57$

Sehingga diperoleh bobot 1 yang baru:

W_1=[162.12 163.06 71.65 0.57]

Iterasi 2:

Data ke-2 [156.39 133.85 40.39 0.58] mewakili buah apel manalagi:

- Jarak ke bobot 1 (jenis buah apel manalagi)

$= \sqrt{(156.39 - 162.12)^2 + (133.85 - 163.06)^2 + (40.39 - 71.65)^2 + (0.58 - 0.57)^2}$

$= 43.03252$

- Jarak ke bobot 2 (jenis buah mangga manalagi)

$= \sqrt{(156.39 - 125.95)^2 + (133.85 - 119.05)^2 + (40.39 - 51.30)^2 + (0.58 - 0.49)^2}$

$= 35.5622$

- Jarak ke bobot 3 (jenis buah jambu merah)

$= \sqrt{(156.39 - 190.65)^2 + (133.85 - 131.45)^2 + (40.39 - 48.83)^2 + (0.58 - 0.54)^2}$

$= 35.29455$

- Jarak ke bobot 4 (jenis buah jeruk manis)

$= \sqrt{(156.39 - 226.10)^2 + (133.85 - 166.99)^2 + (40.39 - 25.08)^2 + (0.58 - 0.59)^2}$

$= 71.41177$

- Jarak ke bobot 5 (jenis buah pisang susu)

$= \sqrt{(156.39 - 201.25)^2 + (133.85 - 132.55)^2 + (40.39 - 38.66)^2 + (0.58 - 0.32)^2}$

$= 44.91292$

Jarak terpendek adalah pada bobot ke-3 dengan target kelas adalah 1, sehingga bobot ke-3 yang baru adalah:

$w_{31} = w_{31} + Lr * (x_{21} - w_{31}) = 190.65 + 0.04 * (156.39 - 190.651) = 184.18$

$w_{32} = w_{32} + Lr * (x_{22} - w_{32}) = 131.45 + 0.04 * (133.85 - 131.45) = 131.61$

$w_{33} = w_{33} + Lr * (x_{23} - w_{33}) = 48.83 + 0.04 * (40.39 - 48.83) = 49.08$

$w_{34} = w_{34} + Lr * (x_{24} - w_{34}) = 0.54 + 0.04 * (0.58 - 0.54) = 0.54$

Sehingga diperoleh bobot 1 yang baru:

W_3=[184.67 122.52 25.94 0.35]

Iterasi 3:

Data ke-3 [121.56 86.24 23.93 0.14] mewakili buah pisang susu:

- Jarak ke bobot 1 (jenis buah apel manalagi)

$$= \sqrt{(184.67 - 162.12)^2 + (122.52 - 163.06)^2 + (25.94 - 71.65)^2 + (0.35 - 0.57)^2}$$
$$= 64.99727$$

- Jarak ke bobot 2 (jenis buah mangga manalagi)

$$= \sqrt{(184.67 - 125.95)^2 + (122.52 - 119.05)^2 + (25.94 - 51.30)^2 + (0.35 - 0.49)^2}$$
$$= 64.05645$$

- Jarak ke bobot 3 (jenis buah jambu merah)

$$= \sqrt{(184.67 - 190.65)^2 + (122.52 - 131.45)^2 + (25.94 - 48.83)^2 + (0.35 - 0.54)^2}$$
$$= 24.86227$$

- Jarak ke bobot 4 (jenis buah jeruk manis)

$$= \sqrt{(184.67 - 226.10)^2 + (122.52 - 166.99)^2 + (25.94 - 25.08)^2 + (0.35 - 0.59)^2}$$
$$= 42.39088$$

- Jarak ke bobot 5 (jenis buah pisang susu)

$$= \sqrt{(184.67 - 201.25)^2 + (122.52 - 132.55)^2 + (25.94 - 38.66)^2 + (0.35 - 0.32)^2}$$
$$= 23.17966$$

Jarak terpendek adalah pada bobot ke-5 dengan target kelas adalah 5, sehingga bobot ke-5 yang baru adalah:

$$w_{51} = w_{51} - Lr * (x_{31} - w_{51}) = 184.67 - 0.04 * (184.67 - 201.25) = 216.17$$
$$w_{52} = w_{52} - Lr * (x_{32} - w_{52}) = 122.52 - 0.04 * (122.52 - 132.55) = 141.58$$
$$w_{53} = w_{53} - Lr * (x_{33} - w_{53}) = 25.94 - 0.04 * (25.94 - 38.66) = 50.12$$
$$w_{54} = w_{54} - Lr * (x_{34} - w_{54}) = 0.35 - 0.04 * (0.35 - 0.32) = 0.29$$

Sehingga diperoleh bobot 5 yang baru:

W_5=[216.17 141.58 50.12 0.29]

Implementasi proses pelatihan LVQ dimulai dengan melakukan inisialisasi input, kelas target, dan parameter-parameter seperti maksimal epoh, *learning rate*, eror minimum yang diharapkan, serta nilai pengurangan *learning rate*. Inisialisasi yang dilakukan pada bahasa C# adalah sebagai berikut:

```
//inisialisi input training
double[,] x = new double[100, 5];

//inisialisi input testing
double[,] tes = new double[1, 4];

//inisialisasi bobot
double[,] w = new double[5,4];
```

```
//inisialisai target
int[] C = { 1, 2, 3, 4, 5 };

//learning rate(alpha)
double lr;

int pilih_lr;
double eps = 0.01;

//alpha
double dec_lr;
int pilih_dec_lr;


//max epoh
int maxepoh;
int epoh = 0;

//euclidean distance
double ed1;
double ed2;
double ed3;
double ed4;
double ed5;
```

Untuk penentuan nilai awal bobot adalah 5 data pertama untuk setiap jenis buah. Pada pengambilan nilai bobot yang dilakukan pada bahasa C# adalah sebagai berikut:

```
int id = 0;

MySqlConnection con = new MySqlConnection(connect_db);

string strSQL = "SELECT * FROM bobot";
MySqlCommand mysqlCmd = new MySqlCommand(strSQL, con
con.Open();
MySqlDataReader reader;
reader = mysqlCmd.ExecuteReader();
while (reader.Read())
{
 w[id, 0] = (double)reader["red"];
 w[id, 1] = (double)reader["green"];
 w[id, 2] = (double)reader["blue"];
 w[id, 3] = (double)reader["roundness"];
 id = id + 1;
}
con.Close();
```

Setelah itu dilakukan pengambilan data *training*. Pengambilan data *training* dari basis data adalah membaca setiap *record* kemudian menyimpannya dalam suatu array. Implementasinya pada bahasa C# dilakukan sebagai berikut:

```
int id = 0;
```

```
MySqlConnection con = new MySqlConnection(connect_db);

string strSQL = "SELECT * FROM fitur";
MySqlCommand mysqlCmd = new MySqlCommand(strSQL, con);
con.Open();
MySqlDataReader reader;
reader = mysqlCmd.ExecuteReader();

while (reader.Read())
{
 x[id, 0] = (double)reader["r"];
 x[id, 1] = (double)reader["g"];
 x[id, 2] = (double)reader["b"];
 x[id, 3] = (double)reader["rd"];

 //load category
 x[id, 4] = (int)reader["j"];

 id = id + 1;
}
con.Close();
```

Dengan telah dilakukan inisialisasi dan pengambilan data *training* dari *database*, maka pelatihan LVQ dapat dilakukan. Pelatihan berlangsung sesuai jumlah maksimal epoh dan *learning rate* seperti berikut ini:

```
while (epoh < maxepoh && lr > eps)
{
 epoh = epoh + 1;
 for (int i = 0; i <= 99; i++)
 {
  train(i);
 }
 lr = lr - (lr * dec_lr);
}
```

Pada fungsi *train()* proses yang terjadi berupa penghitungan jarak *euclidean*, kemudian melakukan perubahan nilai bobot sesuai target kelasnya. Berikut ini adalah proses penghitungan jarak *euclidean* pada bahasa C#:

```
for (int j = 0; j < 4; j++)
 {
  ed1 = ed1 + ((x[a, j] - w[0, j]) * (x[a, j] - w[0, j]));
  ed2 = ed2 + ((x[a, j] - w[1, j]) * (x[a, j] - w[1, j]));
  ed3 = ed3 + ((x[a, j] - w[2, j]) * (x[a, j] - w[2, j]));
  ed4 = ed4 + ((x[a, j] - w[3, j]) * (x[a, j] - w[3, j]));
  ed5 = ed5 + ((x[a, j] - w[4, j]) * (x[a, j] - w[4, j]));
 }

 //akar
 ed1 = Math.Sqrt(ed1);
 ed2 = Math.Sqrt(ed2);
 ed3 = Math.Sqrt(ed3);
 ed4 = Math.Sqrt(ed4);
 ed5 = Math.Sqrt(ed5);
```

Proses perubahan salah satu nilai bobot jika sesuai dengan kelasnya pada bahasa C# adalah sebagai berikut:

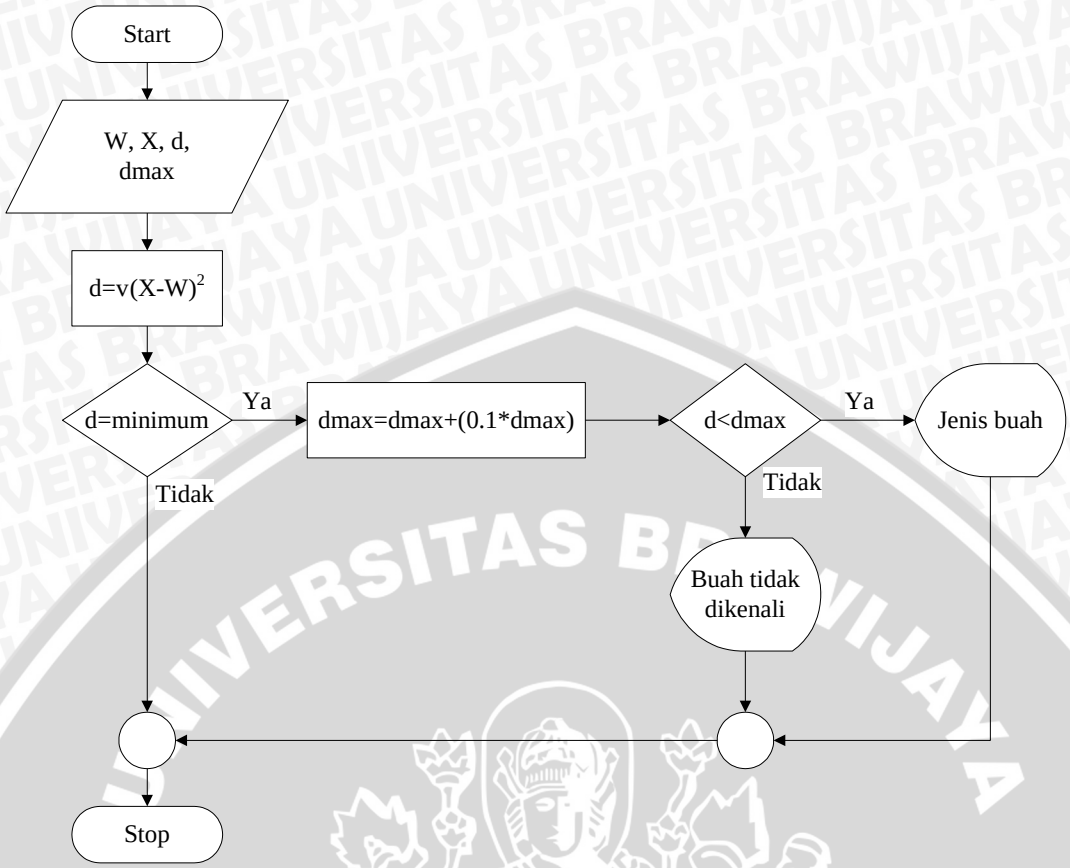
```
//ubah bobot w1
for (int i = 0; i <= 3; i++)
{
 w[0, i] = w[0, i] + lr * (x[a, i] - w[0, i]);
}
con.Open();
string strSQL = "UPDATE bobot SET red='" + w[0,0] + "',green='" + w[0,1] +
"',blue='" + w[0,2] + "',roundness='" + w[0,3] + "' WHERE id_bobot='1'";
MySqlCommand mysqlCmd = new MySqlCommand(strSQL, con);
mysqlCmd.ExecuteNonQuery();
con.Close();
```

Proses perubahan salah satu nilai bobot jika tidak sesuai dengan kelasnya pada bahasa C# adalah sebagai berikut:

```
//ubah bobot w1
for (int i = 0; i <= 3; i++)
{
 w[0,i] = w[0,i] - lr * (x[a, i] - w[0,i]);
}
con.Open();
string strSQL = "UPDATE bobot SET red='" + w[0,0] + "',green='" + w[0,1] +
"',blue='" + w[0,2] + "',roundness='" + w[0,3] + "' WHERE id_bobot='1'";
MySqlCommand mysqlCmd = new MySqlCommand(strSQL, con);
mysqlCmd.ExecuteNonQuery();
con.Close();
```

4.2.6 Perancangan identifikasi buah

Pada tahap perancangan identifikasi buah berdasarkan warna dan bentuk menggunakan webcam dilakukan dengan cara mengukur dan membandingkan jarak suatu vektor input buah yang belum dikettahui jenisnya oleh aplikasi dengan vekotr bobot setiap jenis buah yang telah didapatkan melalui proses pelatihan. Pengukuran jarak dilakukan dengan menggunakan metode euclidean distance. Jarak terkecil vektor buah input yang belum dikenali dengan vektor bobot jenis buah kemudian diperiksa apakah melebihi toleransi 10% jarak maksimum yang didapat dari pelatihan atau tidak. Jika jarak terkecil tersebut melewati nilai maksimum maka buah tidak akan dikenali, hal ini seperti yang dijelaskan pada gambar 4.15.



Gambar 4.15 Flowchart identifikasi buah

(**Sumber:** Perancangan)

Implementasi proses identifikasi buah diawali dengan melakukan deklarasi variabel yang menyimpan perhitungan toleransi jarak. Deklarasi yang dilakukan pada bahasa C# adalah sebagai berikut:

```
double ed1max;
double ed2max;
double ed3max;
double ed4max;
double ed5max;
```

Selanjutnya adalah memanggil data *testing* yang berupa hasil ekstraksi ciri pada fase pengujian yang telah disimpan dalam array. Pemanggilan data *testing* yang terjadi pada bahasa C# adalah sebagai berikut:

```
tes[0, 0] = Ravg;
tes[0, 1] = Gavg;
tes[0, 2] = Bavg;
tes[0, 3] = roundness;
```

Setelah itu melakukan proses perhitungan jarak *euclidean* terhadap nilai bobot yang akan menentukan jenis daripada buah uji. Berikut ini adalah implementasi perhitungan jarak *euclidean* pada bahasa C#:

```
for (int j = 0; j <= 3; j++)
{
ed1 = ed1 + ((tes[a, j] - w[0,j]) * (tes[a, j] - w[0,j]));
ed2 = ed2 + ((tes[a, j] - w[1,j]) * (tes[a, j] - w[1,j]));
ed3 = ed3 + ((tes[a, j] - w[2,j]) * (tes[a, j] - w[2,j]));
ed4 = ed4 + ((tes[a, j] - w[3,j]) * (tes[a, j] - w[3,j]));
ed5 = ed5 + ((tes[a, j] - w[4,j]) * (tes[a, j] - w[4,j]));
}

//akar
ed1 = Math.Sqrt(ed1);
ed2 = Math.Sqrt(ed2);
ed3 = Math.Sqrt(ed3);
ed4 = Math.Sqrt(ed4);
ed5 = Math.Sqrt(ed5);
```

Untuk menentukan jenis buah harus dilakukan suatu pemeriksaan jarak terkecil terhadap salah satu nilai bobot. Selain itu juga masih dilakukan pemeriksaan toleransi terhadap nilai jarak tersebut. Berikut ini adalah implementasi salah satu penentuan jenis buah pada bahasa C#:

```
if (ed1 < ed2 && ed1 < ed3 && ed1 < ed4 && ed1 < ed5)
{
 con.Open();
 string strSQL = "SELECT MAX(ed1) AS ed1max FROM ed1";
 MySqlCommand mysqlCmd = new MySqlCommand(strSQL, con);
 mysqlCmd.ExecuteNonQuery();
 MySqlDataReader reader;
 reader = mysqlCmd.ExecuteReader();

 while (reader.Read())
 {
  ed1max=(double)reader["ed1max"];
 }
 con.Close();

 ed1max = ed1max + (ed1max * 0.1);

 if (ed1 < ed1max)
 {
  textBox1.Text = "APEL MANALAGI";
 }
 else
 {
  textBox1.Text = "BUAH TIDAK DAPAT DIKENALI";
 }
}
```

4.3 Implementasi Sistem

Setelah tahap perancangan tahap selanjutnya adalah tahap implementasi. Implementasi ini merupakan proses transformasi hasil perancangan perangkat lunak yang telah dibuat ke dalam kode *(coding)* sesuai dengan sintaks dari bahasa pemrograman yang digunakan.

4.3.1 Lingkungan Implementasi

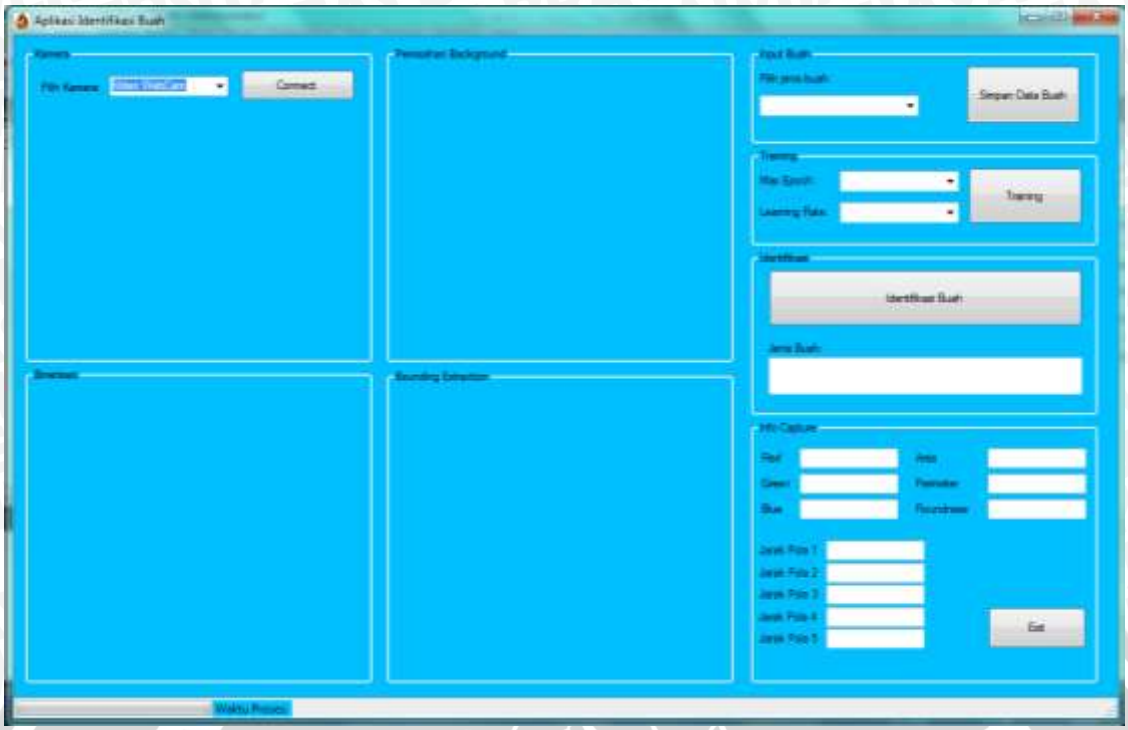
Aplikasi dibuat dengan menggunakan Microsoft Visual Studio C# dengan Aforge .NET framework. Sistem diimplementasikan dengan menggunakan spesifikasi sebagai berikut:

1. Perangkat keras :
   A. Komputer
      Spesifikasi :
      - Processor : AMD Turion™ II X2 processor M520 2.3GHz
      - Memory : 3072 MB RAM
      - OS : Windows 7 Ultimate
      - VGA : ATI Radeon HD 4200 256MB

   B. *Webcam*
      Spesifikasi:
      - Model : i-Tech
      - Resolusi : 1,3 megapiksel
      - Frame/sec : 30 fps
      - Video *Capture* : AVI - 640 x 480, AVI - 320 x 240
2. Perangkat Lunak :
   - Sistem operasi : Microsoft Windows 7 Ultimate
   - Bahasa pemrograman: Microsoft Visual Studio C# .NET 2010
   - Framework : Aforge .NET Framework
   - Basis data : MySQL
   - Tools : XAMPP 1.7.2

4.4 Implementasi Antarmuka

Aplikasi identifikasi buah berdasarkan warna dan bentuk menggunakan webcam memiliki tampilan seperti pada gambar 4.16. Aplikasi ini hanya memiliki 1 tampilan utama yang memiliki beberapa bagian proses sesuai dengan perancangan.



Gambar 4.16 Tampilan aplikasi

(**Sumber:** Perancangan)

Pada gambar 4.16 terdapat 4 groupbox, yang mana setiap groupbox memiliki picturebox. Masing-masing picturebox akan menampilkan hasil setiap proses yang terjadi, dalam hal ini adalah *capture image*, pemisahan *background*, binerisasi, dan *boundary detection*. Sebelum melakukan pengambilan gambar, pengguna diharuskan memilih perangkat webcam yang akan digunakan dan kemudian menghubungkannya dengan aplikasi.

Untuk melakukan penyimpanan data buah, maka pengguna dapat menggunakan fasilitas pada bagian kanan atas yaitu input buah. Bagian selanjutnya adalah training, dimana pengguna dapat melakukan proses pelatihan LVQ dengan mengatur terlebih dahulu nilai *maxepoh* dan *learning rate*. Proses identifikasi dapat dilakukan pada bagian identifikasi. Pada bagian *info capture* ditampilkan beberapa informasi seperti nilai *mean* warna dan *roundness*, serta jarak *euclidean distance*. Pada bagian paling bawah akan dimunculkan waktu proses yang terjadi.

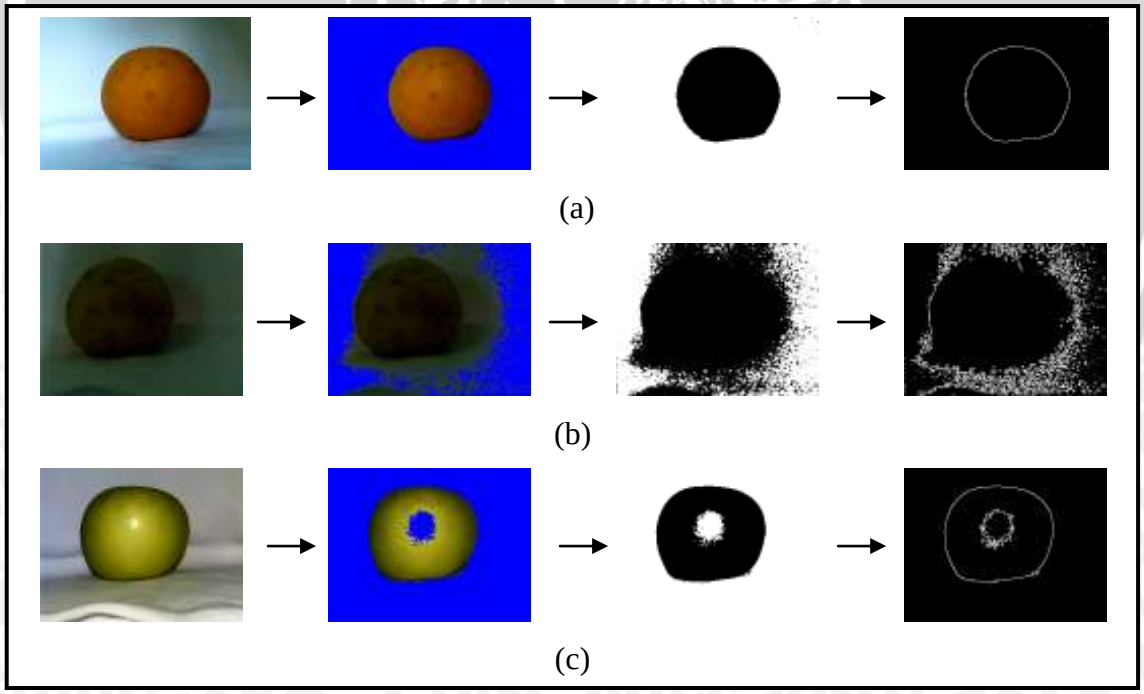
BAB V
PENGUJIAN

Rancangan yang telah diimplementasikan dalam bentuk nyata memerlukan suatu pengujian. Uji coba ini adalah cara untuk mengetahui hasil dari penelitian yang dilakukan sekaligus sebagai sarana pemunculan ide-ide bagi proses pengembangan selanjutnya. Pengujian yang dilakukan dalam bab ini adalah sebagai berikut:

1. Pengujian ekstraksi ciri
2. Pengujian pelatihan LVQ
3. Pengujian identifikasi buah
4. Analisis faktor kegagalan

5.1 Pengujian Ekstraksi Ciri

Pengujian ini bertujuan untuk mengetahui proses pengambilan gambar buah melalui webcam, pemisahan *background* dan ekstraksi ciri telah bekerja sesuai yang diharapkan atau tidak. Gambar 5.1 adalah 3 contoh pengujian ekstraksi ciri.



Gambar 5.1 Contoh ekstraksi ciri dengan 3 kondisi yang berbeda
(a) penerangan yang cukup (b) penerangan yang kurang (c) penerangan yang berlebih
(**Sumber:** Pengujian)

Pada gambar 5.1, terdapat 3 contoh tahapan proses pengambilan gambar, pemisahan *background*, binerisasi dan ekstraksi ciri. Ketiga contoh ini mewakili 3 kondisi berbeda saat pengambilan gambar, yaitu saat penerangan yang cukup, penerangan kurang dan penerangan yang berlebihan. Pengambilan gambar diupayakan dalam kondisi penerangan yang cukup karena jika penerangannya kurang atau berlebihan maka akan menyebabkan eror pada proses selanjutnya.

Hasil ekstraksi ciri pada 100 buah pelatihan dapat dilihat pada tabel 5.1 dan tabel 5.2. Tabel 5.1 menunjukkan ekstraksi ciri warna yaitu *mean* warna *red*, *green*, dan *blue* sedangkan tabel 5.2 menunjukkan ekstraksi ciri bentuk yaitu luas, keliling, dan *roundness*.

Tabel 5.1 Nilai *mean* warna (RGB) pada setiap jenis buah

No.	Jenis Buah	Minimum			Maximum		
		Red	Green	Blue	Red	Green	Blue
1.	Apel manalagi	150.08	123.04	40.39	202.42	171.36	90.24
2.	Mangga manalagi	79.00	71.44	20.42	130.63	127.04	58.15
3.	Jambu merah	160.69	108.52	37.62	193.82	159.24	85.25
4.	Jeruk manis	194.31	112.70	11.04	238.24	152.04	37.09
5.	Pisang susu	180.52	116.16	25.70	209.54	156.71	76.74

(**Sumber:** Pengujian**)**

Tabel 5.2 Nilai luas, keliling dan *roundness* pada setiap jenis buah

No.	Jenis Buah	Luas (Pixel)		Keliling (Pixel)		Roundness	
		Min	Max	Min	Max	Min	Max
1.	Apel manalagi	11380	15315	515	671	0.38	0.58
2.	Mangga manalagi	13960	22865	605	1048	0.25	0.53
3.	Jambu merah	7397	14050	393	584	0.44	0.60
4.	Jeruk manis	11612	17941	501	629	0.52	0.59
5.	Pisang susu	11760	16161	645	1038	0.16	0.39

(**Sumber:** Pengujian)

Dari tabel 5.1 dan 5.2 tampak bahwa terdapat selisih antara nilai maksimum dan nilai minimum untuk setiap cirri buah. Hal ini dikarenakan pada satu jenis buah yang diambil memiliki warna dan bentuk yang berbeda namun tetap pada ciri dari jenis buah tersebut.

5.2 Pengujian pelatihan LVQ

Pengujian ini bertujuan untuk mengetahui pengaruh dari beberapa parameter LVQ seperti maksimal epoh, *learning rate* (rasio pembelajaran), dan pengurangan *learning rate* terhadap waktu yang dibutuhkan pada pelatihan LVQ dan juga tingkat keberhasilan pengenalan. Dengan nilai minimal eror yang diharapkan adalah tetap yaitu 0.01, pengujian pengenalan dilakukan terhadap 25 buah pelatihan dengan 5 buah setiap jenisnya.

5.2.1 Parameter maksimal epoh

Pengujian berdasarkan maksimal epoh dilakukan dengan cara mengganti nilai maksimal epoh yang diinginkan dengan *learning rate* dan pengurangan pembelajaran yang tetap. Nilai *learning rate* adalah 0.9 dan pengurangan *learning rate* adalah 0.01. Nilai maksimal epoh yang diuji adalah 10, 30, dan 50.

Tabel 5.3 Pengujian parameter maksimal epoh pada proses pelatihan LVQ

No.	Maxepoh	Waktu (s)	Keberhasilan Pengenalan
1.	10	17	64%
2.	30	27	76%
3.	50	42	84%

(**Sumber:** Pengujian)

Pengujian berdasarkan maksimal epoh menghasilkan bahwa semakin besar nilai maksimal epoh maka semakin lama waktu yang dibutuhkan untuk pelatihan jaringan LVQ. Hal ini dapat ditunjukkan pada tabel 5.3. Tingkat keberhasilan pengenalan juga semakin tinggi apabila maksimal epoh semakin besar.

5.2.2 Parameter learning rate

Pengujian berdasarkan *learning rate* dilakukan dengan cara mengganti nilai *learning rate* yang diinginkan dengan maxepoh dan pengurangan pembelajaran yang

tetap. Nilai maksimal epoh adalah 50 dan pengurangan pembelajaran adalah 0.1. Nilai learning rate yang diuji adalah 0.3, 0.6, dan 0.9.

Tabel 5.4 Pengujian parameter *learning rate* pada proses pelatihan LVQ

No.	Learning rate	Waktu (s)	Keberhasilan Pengenalan
1.	0.3	28	72%
2.	0.6	31	72%
3.	0.9	33	76%

(**Sumber:** Pengujian)

Pengujian berdasarkan learning rate menghasilkan bahwa semakin besar nilai learning rate maka semakin banyak waktu yang dibutuhkan untuk pelatihan jaringan LVQ denga tingkat keberhasilan pengenalan yang cenderung semakin tinggi. Hal ini dapat ditunjukkan pada tabel 5.4.

5.2.3 Parameter pengurangan *learning rate*

Pengujian berdasarkan pengurangan *learning rate* dilakukan dengan cara mengganti nilai pengurangan pembelajaran yang diinginkan dengan maksimal epoh dan *learning rate* yang tetap. Nilai maksimal epoh adalah 50 dan learning rate adalah 0.9. Nilai pengurangan pembelajaran yang diuji adalah 0.01, 0.05, dan 0.1.

Tabel 5.5 Pengujian parameter pengurangan pembelajaran pada proses pelatihan LVQ

No.	Pengurangan pembelajaran	Waktu (s)	Keberhasilan Pengenalan
1.	0.01	42	84%
2.	0.05	34	76%
3.	0.1	33	76%

(**Sumber:** Pengujian)

Pengujian berdasarkan pengurangan *learning rate* menghasilkan bahwa semakin besar nilai pengurangan *learning rate* maka semakin sedikit waktu yang dibutuhkan untuk pelatihan jaringan LVQ serta tingkat keberhasilan pengenalan semakin kecil. Hal ini dapat ditunjukkan pada tabel 5.5.

5.3 Pengujian identifikasi buah

Pengujian identifikasi ini dilakukan dengan tujuan mengetahui tingkat keberhasilan aplikasi dalam mengenali buah. Sebanyak 23 buah yang akan diuji merupakan buah yang tidak ikut dalam pelatihan yaitu 5 buah apel manalagi, 4 buah mangga manalagi, 3 buah jambu merah, 4 buah jeruk manis, 3 buah pisang susu, 2 buah apel rome beauty dan 2 buah mangga gadung. Dua puluh buah ini berupa buah dengan jenis yang telah ada pada aplikasi dan yang belum ada pada aplikasi.

Dari beberapa parameter maksimal epoh=50, *learning rate*=0.9, dan pengurangan *learning rate*=0.01 didapatkan bahwa aplikasi dapa mengenali 17 buah dengan baik atau memiliki tingkat keberhasilan pengenalan buah sebesar 73.9%. Tabel 5.6 menunjukkan beberapa contoh hasil pengujian identifikasi terhadap buah.

Tabel 5.6 Contoh hasil pengujian identifikasi

No.	Jenis Buah	Hasil Identifikasi
1.	Jeruk manis	Jeruk manis
2.	Apel manalagi	Apel manalagi
3.	Mangga manalagi	Mangga manalagi
4.	Jambu merah	Jambu merah
5.	Pisang susu	Pisang susu
6.	Jambu merah	Apel manalagi
7.	Pisang susu	Pisang susu
8.	Apel manalagi	Apel manalagi
9.	Apel manalagi	Apel manalagi
10.	Apel manalagi	Apel manalagi
11.	Jambu merah	Jambu merah
12.	Jeruk manis	Jeruk manis
13.	Apel manalagi	Apel manalagi
14.	Pisang susu	Jambu merah
15.	Jeruk manis	Jeruk manis
16.	Jeruk manis	Jeruk manis
17.	Mangga manalagi	Mangga manalagi
18.	Mangga manalagi	Mangga manalagi
19.	Apel rome beauty	Apel manalagi

20.	Mangga gadung	Mangga manalagi
21.	Mangga manalagi	Mangga manalagi
22.	Apel rome beauty	Apel manalagi
23.	Mangga gadung	Mangga manalagi

(**Sumber:** Pengujian)

5.4 Analisis faktor kegagalan

Pada kenyataannya aplikasi identifikasi buah berdasarkan warna dan bentuk menggunakan *webcam* ini tidak selalu berjalan secara optimal. Ada beberapa faktor yang dapat menyebabkan proses yang dilakukan menjadi gagal. Berikut adalah beberapa faktor-faktor yang menyebabkan kegagalan tersebut.

1. Intensitas cahaya

   Pencahayaan yang terlalu terang atau terlalu gelap mengakibatkan pengambilan gambar menjadi kurang baik. Hal ini akan menyebabkan proses ekstraksi ciri yang tidak optimal.

2. Webcam

   Kualitas sensor *webcam* yang kurang bagus menyebabkan bila ada cahaya kuat dari lampu penerangan maka pengambilan gambar akan didominasi dengan wana putih sehingga akan terjadi kegagalan pada ekstraksi ciri.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, pengujian dan analisis sistem maka dapat diambil kesimpulan sebagai berikut:

1. Kualitas citra akan mempengaruhi hasil ekstraksi ciri.
2. Tingkat keberhasilan pengenalan buah saat pengujian bergantung pada proses pelatihan LVQ yang melibatkan beberapa parameter yaitu maksimal epoh, *learning rate*, dan pengurangan *learning rate*.
3. Dari 23 buah yang diujikan, 17 buah mampu dikenali dengan baik pada saat pelatihan dengan nilai maksimal epoh=50, *learning rate*=0.9, dan pengurangan *learning rate*=0.01.
4. Hasil identifikasi bergantung pada hasil proses ekstraksi ciri dan pelatihan LVQ.

6.2 Saran

Dalam perancangan dan pembuatan aplikasi identifikasi buah berdasarkan warna dan bentuk menggunakan *webcam* masih terdapat kekurangan dan kelemahan, oleh karena itu diperlukan penyempurnaan untuk pengembangan selanjutnya. Berikut ini adalah beberapa hal yang perlu diperhatikan untuk pengembangan:

1. Penambahan ciri buah dapat menghasilkan identifikasi yang lebih maksimal.
2. Diperlukan suatu metode ekstraksi ciri yang lebih baik yang lebih mampu untuk membedakan ciri suatu buah.
3. Penggunaan metode lain dalam pengambilan keputusan agar dapat dibandingkan keberhasilan dalam identifikasi dengan metode LVQ.

DAFTAR PUSTAKA

Anonim. Buah. http://id.wikipedia.org/wiki/Buah (diakses tanggal 20 Oktober 2011)

Costa, Luciano da Fontoura and Cesar, Roberto M. Jr. 2001, *Shape and Analysis Classification.* New York: CRC Press LLC.

Elista. 2008. Jaringan Syaraf Tiruan. Semarang: Universitas Diponegoro

Emy. 2007. Morfologi Citra. Yogyakarta: Institut Sains dan Teknologi AKPRIND

Fausett, Laurene. 1993. *Fundamentals of Neural Networks*. New Jersey: Prentice-Hall

Gonzales, Rafael C. and Woods, Richard E. 2002. *Digital Image Processing 2nd Edition*. New Jersey: Prentice-Hall

Kirilov, Andrew. Image Processing and Computer Vision. http://aforgenet.com/forum/viewforum.php?f=4&sid=70eaae4c47de1d0ecbf8736a6bf8c935 (Diakses tanggal 11 September 2011).

Mohsenin, Nuri N. 1986. *Physical Properties of Plant and Animal Materials*. New York: Gordon and Briech Science Publishers

Powell, Gavin. 2006. *Beginning Database Design*. Indianapolis: Wiley Publishing, Inc.

Putra, Darma. 2010. Pengolahan Citra Digital. Yogyakarta: Andi

Ranadhi, Djalu dkk. 2006. Implementasi Learning Vector Quantization (LVQ) Untuk Pengenal Pola Sidik Jari Pada Sistem Informasi Narapidana Lp Wirogunan. Yogyakarta: Jurusan Teknik Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia.

Seng, Woo C. and Mirisaee, Seyed H. *A New Method for Fruits Recognition System*, Malaysia.

Suhendra, Adang. Catatan Kuliah Pengantar Pengolahan Citra. Jakarta: Universitas Gunadarma.

Talibo, Lukman. 2004. Sistem Pengenalan Obyek Real-Time Dengan Jaringan Syaraf Tiruan Backpropagation. Bandung: Teknik Informatika UNIKOM

LAMPIRAN

Lampiran 1. Citra buah untuk pelatihan

Lampiran 2. Citra buah untuk pengujian