BAB II
TINJAUAN PUSTAKA

1.1 Microsoft Visual Studio

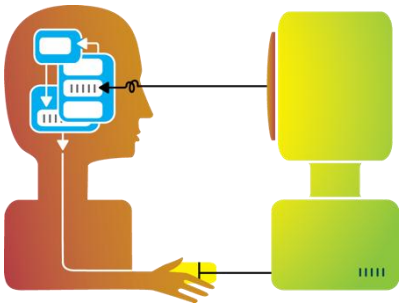
Microsoft *Visual Studio* merupakan sebuah perangkat lunak lengkap (*suite*) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi *console*, aplikasi *Windows*, ataupun aplikasi *Web*. *Visual Studio* mencakup kompiler, SDK, *Integrated Development Environment* (IDE), dan dokumentasi (umumnya berupa MSDN *Library*). Kompiler yang dimasukkan ke dalam paket *Visual Studio* antara lain *Visual* C++, *Visual* C#, *Visual Basic*, *Visual Basic* .NET, *Visual InterDev*, *Visual* J++, *Visual* J#, *Visual FoxPro*, *dan* *Visual SourceSafe*.

Microsoft *Visual Studio* dapat digunakan untuk mengembangkan aplikasi dalam *native code* (dalam bentuk bahasa mesin yang berjalan di atas Windows) ataupun *managed code* (dalam bentuk *Microsoft Intermediate Language* di atas NET *Framework).* Selain itu, *Visual Studio* juga dapat digunakan untuk mengembangkan aplikasi *Silverlight*, aplikasi *Windows Mobile* (yang berjalan di atas .NET *Compact Framework*).[5]

Gambar 1.1 Logo *Micrrosoft Visual Studio*[5]

1.2 Human Computer Interaction (HCI)

Human Computer Interaction (HCI) atau interaksi manusia dan komputer ialah disiplin ilmu yang mempelajari suatu hubungan antara manusia serta komputer yang meliputi perancangan, evaluasi, serta implementasi antarmuka pengguna komputer agar mudah digunakan oleh manusia. Sedangkan interaksi manusia dan komputer itu sendiri ialah serangkaian proses, dialog serta kegiatan yang dilakukan oleh manusia untuk berinteraksi dengan komputer secara interaktif untuk dapat melaksanakan serta menyelesaikan tugas yang diinginkan.



Gambar 1.2 HCI [6]

Interaksi manusia serta komputer ialah suatu ilmu yang sangat berkaitan dengan disain implementasi serta evaluasi dari sistem komputasi yang interaktif untuk dapat digunakan oleh manusia serta studi tentang ruang lingkupnya, ada interaksi antara satu ataupun lebih manusia serta satu atau lebih komputasi mesin. Agar komputer dapat diterima secara luas serta digunakan secara efektif, maka perlu dirancang secara baik.

Hal tersebut tidak berarti bahwa semua sistem harus dirancang agar dapat mengakomodasi semua orang, tetapi komputer perlu dirancang agar memenuhi serta mempunyai kemampuan sesuai dengan kebutuhan pengguna secara spesifik. Tahun 1970 mulailah dikenal istilah antarmuka pengguna (*user interface*), yang juga dikenal dengan istilah sebagai *Man-Machine Interface* (MMI), serta mulai menjadi topik perhatian bagi peneliti dan perancang sistem. Perusahaan komputer

mulai memikirkan aspek fisik dari antarmuka pengguna sebagai faktor untuk penentu keberhasilan dalam pemasaran produknya.

Istilah *Human Computer Interaction* (HCI) mulai muncul pada pertengahan tahun 1980-an sebagai bidang studi yang baru. Istilah HCI juga mengisyaratkan bahwa bidang studi ini mempunyai fokus yang lebih luas, tidak hanya pada perancangan antarmuka secara fisik. HCI didefinisikan ialah sebagai disiplin ilmu yang berhubungan dengan perancangan, evaluasi, serta implementasi sistem komputer interaktif untuk dapat digunakan oleh manusia serta studi tentang fenomena di sekitarnya. HCI pada prinsipnya membuat agar sistem dapat berdialog dengan penggunanya seramah mungkin (*user friendly*). Tidak hanya pada perancangan *layout* layar monitor.[6]

1.3 Webcam

Webcam merupakan singkatan dari web dan camera adalah sebutan bagi kamera waktu-nyata yang gambarnya bisa dilihat melalui *Waring Wera Wanua*, program pengolah pesan cepat, atau aplikasi pemanggilan video. Istilah *webcam* merujuk pada teknologi secara umumnya, sehingga kata *webcam* kadang-kadang diganti dengan kata lain yang memberikan pemandangan yang ditampilkan di kamera. *Webcam* adalah sebuah kamera video digital kecil yang dihubungkan ke komputer melalui colokan USB atau pun colokan COM.

Gambar 1.3 *Webcam* [7]

Fungsi dari *webcam* telah kita ketahui yaitu untuk memudahkan kita dalam mengolah pesan cepat seperti chat melaui video atau bertatap muka melalui video secara langsung. *Webcam* juga berfungsi sebagai alat untuk mentransfer sebuah media secara langsung, namun perlu di sadari kebanyakan pengguna menggunakan piranti ini hanya untuk *chat* video.

Sebuah *webcam* yang sederhana terdiri dari sebuah lensa standar, dipasang di sebuah papan sirkuit untuk menangkap sinyal gambar; *casing* (*cover*), termasuk *casing* depan dan *casing* samping untuk menutupi lensa standar dan memiliki sebuah lubang lensa di *casing* depan yang berguna untuk memasukkan gambar; kabel *support*, yang dibuat dari bahan yang fleksibel, salah satu ujungnya dihubungkan dengan papan sirkuit dan ujung satu lagi memiliki *connector*, kabel ini dikontrol untuk menyesuaikan ketinggian, arah dan sudut pandang web camera. Sebuah *webcam* biasanya dilengkapi dengan *software*, *software* ini mengambil gambar-gambar dari kamera digital secara terus menerus ataupun dalam interval waktu tertentu dan menyiarkannya melalui koneksi internet. Ada beberapa metode penyiaran, metode yang paling umum adalah *hardware* mengubah gambar ke dalam bentuk file JPG dan memasukkannya ke *web server* menggunakan *File Transfer Protocol* (FTP).

Frame rate mengindikasikan jumlah gambar sebuah *software* dapat ambil dan *transfer* dalam satu detik. Untuk *streaming* video, dibutuhkan minimal 15 *frame per second* (fps) atau idealnya 30 fps. Untuk mendapatkan *frame rate* yang tinggi, dibutuhkan koneksi internet yang tinggi kecepatannya. Sebuah *webcam* tidak harus selalu terhubung dengan komputer, ada *webcam* yang memiliki *software webcam* dan *web server bulit-in*, sehingga yang diperlukan hanyalah koneksi internet. *Webcam* seperti ini dinamakan "*network camera*". Kita juga bisa menghindari penggunaan kabel dengan menggunakan hubungan radio, koneksi *Ethernet* ataupun *WiFi*.[7]

1.4 Computer Vision

Computer Vision merupakan suatu proses otomatis yang mengitegrasikan sejumlah besar proses untuk persepsi awal, seperti akuisisi citra, pengolahn citra, klasifikasi, pengenalan (*recognition*) dan membuat keputusan. *Computer Vision* mencoba meniru bagaimana cara kerja sistem *visual* manusia (*Human Vision*) yang sebenarnya sangat kompleks. Objek atau citra yang dilihat ditangkap oleh manusia dengan indera penglihatan kemudian dilanjutkan ke otak untuk diinterpretasi sehingga manusia dapat mengerti objek apa yang terlihat dalam pandangan penglihatannya. Hasil interpretasi ini dapat dipakai untuk pengambilan keputusan (misal menghindar jika melihat ada mobil didepan).

Computer Vision didefinsikan sebagai salah satu cabang ilmu pengetahuan yang mempelajari bagaimana komputer dapat mengenali objek yang diamati atau diobservasi. Cabang ilmu ini bersama Intelejensia Semu (*Artificial Intelegence*) akan mampu menghasilkan sistem Intelijen Visual (*Visual Intellegence System*). Perbedaannya adalah *Computer Vision* lebih mempelajari bagaimana proses komputer dalam mengobservasi suatu objek. Namun komputer grafika lebih kearah pemanipulasian gambar (*visual*) secara digital. Bentuk sederhana dari grafika komputer adalah grafika komputer 2D dan kemudian berkembang menjadi grafika 3D. Pemprosesan citra (*image processing*), dan pengenalan pola (*pattern recognition*).

Computer vision adalah kombinasi antara pengolahan citra dan pengenalan pola. Pengolahan citra (*image processing*) merupakan bidang yang berhubungan dengan proses transformasi citra atau gambar (*image*). Proses ini bertujuan untuk mendapatkan kualitas citra yang lebih baik. Sedangkan pengenalan pola (*pattern recognition*), bidang ini berhubungan denga proses identifikasi objek pada citra atau interpretasi citra. Proses ini bertujuan untuk mengekstrak informasi atau pesan yang disampaikan oleh suatu citra.

Computer Vision diaplikasikan secara beragam, mulai dari *vision system* untuk mesin industri sebagai pemeriksa botol selama produksi, sampai kepada riset untuk kecerdasan buatan menggunakan komputer ataupun robot agar dapat mengenali lingkungan sekitar. Bidang *Computer Vision* meliputi teknologi analisis citra secara otomatis yang digunakan dibanyak bidang. *Machine vision* biasanya mengacu pada proses yang menggabungkan analisis citra secara otomatis dengan metode dan teknologi pengamatan otomatis dan bimbingan robot untuk aplikasi industri.

Dibidang *sains, Computer Vision* berhubungan dengan teori dari sistem-sistem buatan yang dapat mengekstrak informasi dari citra-citra. Contohnya dari rangkaian video, citra, dari beberapa kamera, atau data multi dimensi dari pemindai medis. Di bidang teknologi, *Computer Vision* berusaha untuk menerapkan teori-teori dan model-model untuk pembangunan sistem dari *Computer Vision.*[8]

1.5 Bahasa Pemprograman C#

C# adalah bahasa pemrograman baru yang diciptakan oleh Microsoft yang dikembangkan dibawah kepemimpinan Anders Hejlsberg yang telah menciptakan berbagai macam bahasa pemrograman termasuk Borland Turbo C++ dan Orland Delphi. Bahasa C# juga telah di standarisasi secara internasional oleh ECMA. Seperti halnya bahasa pemrograman yang lain, C# bisa digunakan untuk membangun berbagai macam jenis aplikasi, seperti aplikasi berbasis *windows* (*desktop*) dan aplikasi berbasis *web* serta aplikasi berbasis *web services*.[9]

1.5.1 Sejarah Bahasa Pemprograman C#

Pada akhir dekade 1990-an, *Microsoft* membuat program *Microsoft Visual* J++ sebagai sebuah langkah percobaan untuk menggunakan Java di dalam sistem operasi *Windows* untuk meningkatkan antarmuka dari *Microsoft Component Object Model* (COM). Akan tetapi, akibat masalah dengan pemegang hak cipta

bahasa pemrograman Java, *Sun Microsystems*, *Microsoft* pun menghentikan pengembangan J++, dan beralih untuk membuat pengganti J++, kompilernya dan mesin virtualnya sendiri dengan menggunakan sebuah bahasa pemrograman yang bersifat *general-purpose*.

Untuk menangani proyek ini, *Microsoft* merekrut Anders Helsberg, yang merupakan mantan karyawan Borland yang membuat bahasa Turbo Pascal, dan Borland Delphi, yang juga mendesain *Windows Foundation Classes* (WFC) yang digunakan di dalam J++. Sebagai hasil dari usaha tersebut, C# pun pertama kali diperkenalkan pada bulan Juli 2000 sebagai sebuah bahasa pemrograman modern berorientasi objek yang menjadi sebuah bahasa pemrograman utama di dalam pengembangan di dalam platform *Microsoft.NET Framework*. Pengalaman Helsberg sebelumnya dalam pendesain bahasa pemrograman seperti Visual J++, Delphi, Turbo Pascal) dengan mudah dilihat dalam sintaksis bahasa C#, begitu pula halnya 3 pada inti *Common Language Runtime* (CLR). Dari kutipan atas *interview* dan makalah-makalah teknisnya ia menyebutkan kelemahan-kelemahan yang terdapat pada bahasa pemrograman yang umum digunakan saat ini, misalnya C++, Java, Delphi, ataupun *Smalltalk*. Kelemahan-kelemahan yang dikemukakannya itu yang menjadi basis CLR sebagai bentukan baru yang menutupi kelemahan-kelemahan tersebut, dan pada akhirnya mempengaruhi desain pada bahasa C# itu sendiri.

Ada kritik yang menyatakan C# sebagai bahasa yang berbagi akar dari bahasa-bahasa pemrograman lain. Fitur-fitur yang diambilnya dari bahasa C++ dan Java adalah desain berorientasi objek, seperti *garbage collection*, *reflection*, akar kelas (*root class*), dan juga penyederhanaan terhadap pewarisan jamak (*multiple inheritance*). Fitur-fitur tersebut di dalam C# kini telah diaplikasikan terhadap iterasi, properti, kejadian (*event*), metadata, dan konversi antara tipe-tipe sederhana dan juga objek.

C# didesain untuk memenuhi kebutuhan akan sintaksis C++ yang lebih ringkas dan *Rapid Application Development* yang ‘tanpa batas’ (dibandingkan dengan RAD yang ‘terbatas’ seperti yang terdapat pada Delphi dan *Visual Basic*). Agar mampu mempromosikan penggunaan besar-besaran dari bahasa C#, *Microsoft*, dengan dukungan dari *Intel Corporation* dan Hewlett-Packard, mencoba mengajukan standardisasi terhadap bahasa C#. Akhirnya, pada bulan Desember 2001, standar pertama pun diterima oleh *European Computer Manufacturers Association* atau *Ecma International* (ECMA), dengan nomor standar ECMA-334. Pada Desember 2002, standar kedua pun diadopsi oleh ECMA, dan tiga bulan kemudian diterima oleh *International Organization for Standardization* (ISO), dengan nomor standar ISO/IEC 23270:2006.

C# kadang-kadang dapat disebutkan sebagai bahasa pemrograman yang paling mencerminkan dasar dari CLR dimana semua program-program .NET berjalan, dan bahasa ini sangat bergantung pada kerangka tersebut sebab ia secara spesifik didisain untuk mengambil manfaat dari fitur-fitur yang tersedia pada CLR.[10]

1.5.2 Kelebihan Bahasa Pemprograman C#

Bahasa pemrograman C# memiliki kelebihan yang menonjol dibandingkan dengan bahasa pemrograman yang lainnya. Diantaranya:

1. Bahasa pemrograman C# dibuat sebagai bahasa pemrograman yang bersifat bahasa pemrograman *general-purpose* (untuk tujuan jamak).
2. Berorientasi objek (Object-Oriented Language).
3. Modern.
4. Sederhana (*simple*).
5. *Powerfull* dan fleksibel.
6. Efisien.
7. Modular.
8. C# akan menjadi populer.

1.6 Pengolahan Citra

Pengolahan citra atau *image processing* adalah suatu proses yang dilakukan oleh suatu sistem pada masukan (*input*) berupa citra (*image*) yang mengasilkan (*output*) sehingga menjadi citra (*image*) yang kualitasnya lebih baik. Pengolahan citra bertujuan untuk memperbaiki kualitas citra agar mudah diinterpretasikan oleh manusia atau mesin (dalam hal ini komputer). Namun dengan berkembangnya dunia komputasi yang ditandai dengan semakin meningkatnya kapasitas dan kecepatan proses komputer, serta munculnya ilmu-ilmu komputer yang memungkinkan manusia dapat mengambil informasi dari suatu citra maka *image processing* tidak dapat dilepaskan dengan bidang *computer vision*.[11]

1.6.1 Citra

Citra atau *image* adalah kombinasi antara titik, garis, bidang, dan warna untuk menciptakan suatu imitasi dari suatu objek, biasanya berupa objek fisik atau manusia. Citra dapat berwujud gambar (*image*) dua dimensi, seperti lukisan, foto, dan juga dapat berwujud tiga dimensi, seperti patung. Dalam tinjauan matematis, citra merupakan fungsi kontinyu dari intensitas cahaya pada bidang yang dapat disimpan dalam bentuk atau format digital. Citra tersebut oleh sebuah komputer didefinisikan sebagai sebuah bidang (dengan luas tertentu) yang terdiri dari banyak titik dengan koordinat tertentu. Banyaknya titik dalam bidang inilah yang dinamakan *pixel*. Semakin banyak *pixel*, citra yang dihasilkan semakin jelas.[11]

1.6.2 Model Citra

Citra terdiri dari citra kontinyu dan citra diskrit. Citra kontinyu dihasilkan dari sistem optik yang menerima sinyal analog, misalnya mata manusia dan kamera analog. Citra diskrit dihasilkan melalui proses digitalisasi terhadap citra kontinyu. Beberapa sistem optik dilengkapi dengan fungsi digitalisasi sehingga ia mampu menghasilkan citra diskrit, misalnya kamera digital dan scanner. Citra diskrit disebut juga citra digital.

Komputer digital yang umum dipakai saat ini hanya dapat mengolah citra digital. Citra digital merupakan suatu matriks dimana indeks baris dan kolomnya menyatakan suatu titik pada citra tersebut dan elemen matriksnya (yang disebut sebagai elemen gambar/ *pixel*/ piksel/ pels/ *picture element*) menyatakan tingkat keabuan pada titik tersebut. Piksel sendiri memiliki dua parameter yaitu koordinat dan intensitas atau warna. Nilai yang terdapat pada titik (x,y) adalah f(x,y), yaitu besar intensitas atau warna dari piksel di titik itu. Oleh sebab itu citra digital dapat ditulis dalaam bentuk gambar sebagai berikut.

$$f(x,y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & f(1,1) & \dots & f(1,M-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix}$$

Gambar 1.4 Matriks Citra Digital

Berdasarkan gambaran tersebut, secara matematis citra digital dapat dituliskan sebagaifungsi intensitas f(x,y), dimana harga x(baris) dan (kolom) merupakan koordinat posisi dan f(x,y) adalah nilai fungsi pada setiap titik (x,y) yang menyatakan besar intensitas citra atau tingkat keabuan atau warna dari piksel dititik tersebut. Pada proses digitalisasi diperoleh besar baris ***M*** dan kolom ***N*** hingga citra membentuk matrik ***M x N*** dan jumlah tingkat keabuan piksel G. [11]

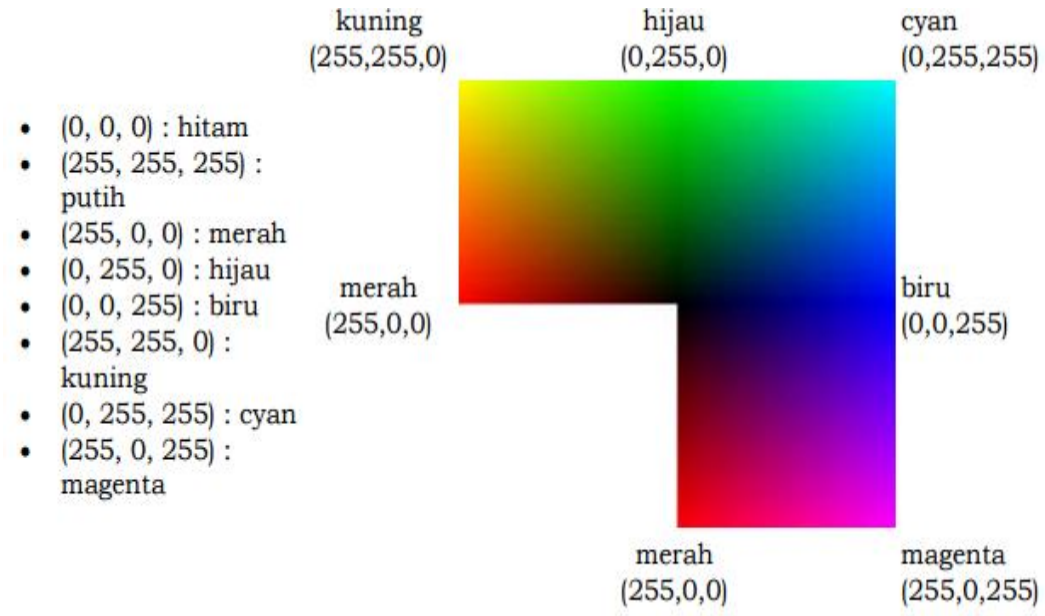
1.6.3 RGB

RGB sering digunakan didalam sebagian besar aplikasi komputer karena dengan ruang warna ini, tidak diperlukan transformasi untuk menampilkan informasi di layar monitor. Alasan diatas juga menyebabkan RGB banyak dimanfaatkan sebagai ruang warna dasar bagi sebagian besar aplikasi. Model warna RGB adalah model warna berdasarkan konsep penambahan kuat cahaya primer yaitu *Red, Green* dan *Blue*. Dalam suatu ruang yang sama sekali tidak ada cahaya, maka ruangan tersebut adalah gelap total. Tidak ada signal gelombang

cahaya yang diserap oleh mata kita atau RGB (0,0,0). Apabila ditambahkan cahaya merah pada ruangan tersebut, maka ruangan akan berubah warna menjadi merah misalnya RGB (255,0,0), semua benda dalam ruangan tersebut hanya dapat terlihat berwarna merah. Demikian juga apabila cahaya diganti dengan hijau atau biru.

Gambar 1.5 Warna RGB [12]

Apabila diberikan 2 macam cahaya primer dalam ruangan tersebut seperti (merah dan hijau), atau (merah dan biru) atau (hijau dan biru), maka ruangan akan berubah warna masing-masing menjadi kuning, atau magenta atau cyan. Warna-warna yang dibentuk oleh kombinasi dua macam cahaya tersebut disebut warna sekunder. Warna Tersier adalah warna yang hanya dapat terlihat apabila ada tiga cahaya primer, jadi apabila dinon-aktifkan salah satu cahaya, maka benda tersebut berubah warna. Contoh warna tersier seperti abu-abu, putih. Pada perhitungan dalam program-program komputer model warna direpresentasi dengan nilai komponennya, seperti dalam RGB (r, g, b) masing-masing nilai antara 0 hingga 255 sesuai dengan urutan masing-masing yaitu pertama *Red*, kedua *Green* dan ketiga adalah nilai *Blue* dengan demikian masing-masing komponen ada 256 tingkat. Apabila dikombinasikan maka ada 256 x 256 x 256 atau 16.777.216 kombinasi warna RGB yang dapat dibentuk.[12]



Gambar 1.6 Konfigurasi RGB [12]

1.6.4 CMYK

CMYK (Cyan, Magenta, Yellow, Key) adalah model warna yang biasanya digunakan di percetakan (Printer, Sablon, dll). Tinta *process cyan*, *process magenta*, *process yellow*, *process black* dicampurkan dengan komposisi tertentu dan tepat serta akurat sehingga menghasilkan warna cetak yang tepat seperti yang diinginkan pada background putih dengan media kertas maupun lainnya. Bahkan bila suatu saat diperlukan, warna ini dengan mudah bisa dibentuk kembali.[13]

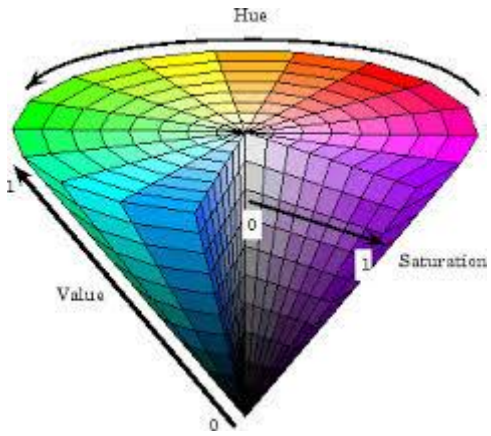
Gambar 1.7 Model CMYK [13]

1.6.5 Colour Filtering

Colour filtering adalah metode yang berguna untuk menemukan sebuah warna yang terdapat pada sebuah citra atau gambar, kita dapat menentukan proses apa yang selanjutnya harus dilakukan. Pada dasarnya pencarian ini menggunakan kombinasi dari komponen RGB yang terdapat pada citra. kombinasi RGB kemudian dijadikan filter untuk kemudian diproses oleh komputer.[14]

1.6.6 Ruang Warna HSV

Model warna HSV mendefinisikan warna dalam terminologi *Hue*, *Saturation* dan *Value*. *Hue* menyatakan warna sebenarnya, seperti merah, violet, dan kuning. *Hue* digunakan untuk membedakan warna-warna dan menentukan kemerahan (*redness*), kehijauan (*greeness*), dsb, dari cahaya. *Hue* berasosiasi dengan panjang gelombang cahaya. *Saturation* menyatakan tingkat kemurnian suatu warna, yaitu mengindikasikan seberapa banyak warna putih diberikan pada warna. *Value* adalah atribut yang menyatakan banyaknya cahaya yang diterima oleh mata tanpa memperdulikan warna.[12]

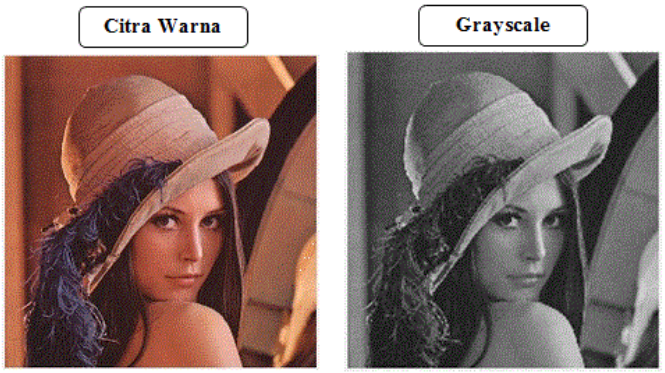


Gambar 1.8 Ruang Warna HSV [12]

1.6.7 Grayscale

Grayscale atau derajat keabuan merupakan proses awal dalam *image processing* yang merupakan suatu metode dalam sistem yang dibangun untuk dilakukan konversi citra dari berbagai format warna seperti RGB, BGR, RGBA, YCrCb, HSV. Tujuannya adalah untuk melakukan pencarian batas atau *edge* atau

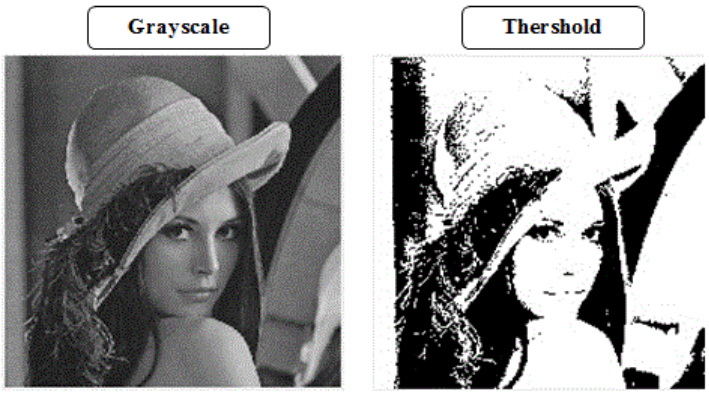
tepi citra lalu menggambarkan tepi-tepi tersebut sehingga komputer akan dapat dengan mudah melihat bentuk atau kontur-kontur yang terdapat dengan citra. Setelah komputer dapat melihat bentuk tersebut maka komputer dapat dengan mudah melakukan berbagai proses komputasi terutama yang berhubungan dengan *recognizing, tracking,* dan *motion* pada suatu gambar bergerak baik itu dari *disk* atau pun *real-time* dengan sumber *input* menggunakan kamera atau *webcam*.[15]



Gambar 1.9 Proses *Grayscale* [15]

1.6.8 Threshold

Threshold merupakan konversi citra hitam putih ke citra biner dilakukan dengan cara mengelompokkan nilai derajat keabuan setiap pixel kedalam 2 kelas, hitam dan putih. Pada citra hitam putih terdapat 256 level, artinya mempunyai skala “0” sampai “255” atau [0,255], dalam hal ini nilai intensitas 0 menyatakan hitam, dan nilai intensitas 255 menyatakan putih, dan nilai antara 0 sampai 255 menyatakan warna keabuan yang terletak antara hitam dan putih.[15]



Gambar 1.10 Proses *Threshold* [15]

1.6.9 Segmentasi Citra

Segmentasi citra merupakan teknik untuk memisahkan atau mengisolasi suatu objek atau sebagian dari objek dari suatu *image* (memisahkan *foreground* dengan *background*). Banyak situasi dalam *computer vision* yang membutuhkan segmentasi atau memisahkan *foreground* atau suatu objek terhadap *background*-nya guna melihat aktivitas *pixel* dari objek yang ingin kita lihat. Misalkan ketika suatu kamera keamanan dilengkapi pendeteksi pergerakan manusia atau objek lainnya yang menyerupai manusia dan beberapa binatang, maka kamera tersebut harus terlebih dahulu dapat melakukan segmentasi terhadap objek manusia ataupun binatang tertentu.

Segmentasi biasa dilakukan dengan maksud agar pencarian *pixel* tidak terlalu memakan waktu dan *memory*, sehingga ketika terjadi komputasi pencarian tidak memerlukan pencarian secara menyeluruh pada suatu citra, cukup pencarian pada daerah yang biasa disebut *region of interest* sehingga dapat menghemat waktu dan *memory*. Pencarian yang dimaksud biasanya berupa proses *tracking* dan *motion* atau pergerakan suatu pixel. Komputasi pendeteksian *motion* ini akan terjadi hanya pada setiap *point* yang telah di segmentasi atau bisa disebut pada objek yang telah dispesifikkan. Ada banyak metode atau algoritma yang dapat digunakan dalam segmentasi bahkan dengan algoritma *image processing* biasa seperti *morphology*, *flood fill*, *threshold*, dan *pyramid* akan tetapi terdapat algoritma tersendiri yang dapat menghasilkan segmentasi yang cukup baik terutama dalam segmentasi warna yang dipakai dalam penelitian ini.

Dalam pembuatan kontur diperlukan rutin untuk melakukan proses pencarian kontur. Citra yang dipakai adalah citra hasil konversi menjadi *grayscale* dan dikenakan *threshold* sehingga menghasilkan citra *bi-level*.[14]

1.6.10 Erosi

Erosi merupakan proses penghapusan titik-titik batas objek menjadi bagian dari latar, berdasarkan *structuring element* yang digunakan. Pada operasi ini, ukuran objek diperkecil dengan mengikis sekeliling objek. Cara yang dapat dilakukan ada 2:

1. Dengan mengubah semua titik batas menjadi titik latar.
2. Dengan menset semua titik di sekeliling titik latar menjadi titik latar.

Untuk semua titik dalam citra
Cek apakah tersebut titik latar
→ Jika ya maka ubah semua tetangganya menjadi titik latar
→ Jika tidak maka lanjutkan

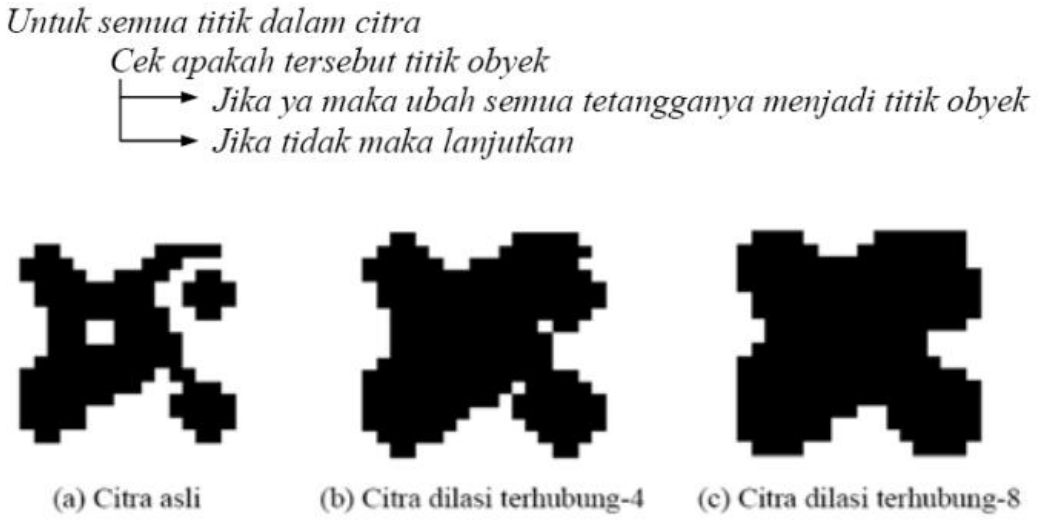
(a) Citra asli (b) Citra erosi terhubung-4 (c) Citra erosi terhubung-8

Gambar 1.11 Proses Erosi [17]

1.6.11 Dilasi

Dilasi merupakan proses penggabungan titik-titik latar menjadi bagian dari objek, berdasarkan *structuring element* yang digunakan. Proses ini adalah kebalikan dari erosi, yaitu merubah latar disekeliling objek menjadi bagian dari objek tersebut. Terdapat 2 cara untuk melakukan operasi ini, yaitu:

1. Dengan cara mengubah semua titik latar yang bertetangga dengan titik batas menjadi titik objek, atau lebih mudahnya *set* setiap titik yang tetangganya adalah titik objek menjadi titik objek.
2. Dengan mengubah semua titik di sekeliling titik batas menjadi titik objek, atau lebih mudahnya *set* semua titik tetangga sebuah titik objek menjadi titik objek.



Gambar 1.12 Proses Dilasi [17]

1.7 Motion dan Tracking

Motion tracking, motion capture atau mocap adalah terminologi yang digunakan untuk mendeskripsikan proses dari perekaman gerakan dan pengartian gerakan tersebut menjadi model digital.[18]

1.8 Optical Flow

Optical Flow adalah perkiraan gerakan suatu bagian dari sebuah citra berdasarkan turunan intensitas cahayanya pada sebuah sekuen citra. Pada ruang 2D hal ini berarti seberapa jauh suatu piksel citra berpindah diantara dua *frame* citra yang berurutan. Sedangkan pada ruang 3D hal ini berarti seberapa jauh suatu volume piksel (voxel) berpindah pada dua volume yang berurutan. Perhitungan turunan dilakukan berdasarkan perubahan intensitas cahaya pada kedua *frame* citra maupun volume. Perubahan intensitas cahaya pada suatu bagian citra dapat disebabkan oleh gerakan yang dilakukan oleh objek, gerakan sumber cahaya, ataupun perubahan sudut pandang.[4]

1.8.1 Algoritma Lucas Kanade

Algoritma Lucas-Kanade, pertama kali diajukan pada tahun 1981, awalnya adalah sebuah usaha untuk mencari teknik registrasi citra yang cepat dengan memanfaatkan *gradient* intensitas spasial. Pada perkembangannya, algoritma ini kemudian menjadi salah satu algoritma *optical flow* yang penting. Berbeda dengan algoritma Horn-Schunk yang bekerja berbasis pada keseluruhan citra, algoritma ini bekerja berdasar pada informasi lokal yang diturunkan dari *window* kecil (*patch*) disekeliling titik yang diperhitungkan. Kelemahan digunakan *window* lokal kecil pada algoritma Lucas-Kanade adalah tidak terdeteksinya gerakan-gerakan yang besar karena gerakan-gerakan tersebut jatuh diluar *window*. Permasalahan ini kemudian dapat diatasi dengan mengimplementasikan penyelesaian dengan prinsip piramida, yaitu *pyramidal* Lucas-Kanade. Prinsip ini merupakan penyelesaian berdasarkan iterasi dari level detil citra paling rendah hingga level detil citra paling tinggi.[4]

1.8.2 Pyramidal Lucas Kanade

Penyelesaian algoritma Lucas-Kanade dengan pendekatan piramida, atau disebut *Pyramidal* Lucas Kanade diajukan pertama kali oleh Jean-Yves Bouguet (Bouguet, 2000). Pendekatan ini menggunakan prinsip piramida, yaitu bekerja dari detil citra paling rendah hingga detil citra paling tinggi. Tujuannya adalah agar gerakan yang “besar” dapat diperhitungkan. Sementara asumsi yang digunakan pada algoritma Lucas-Kanade adalah gerakan yang “kecil” dan koheren, sehingga tidak dapat menangkap gerakan yang “besar”. Solusi untuk dapat menangkap gerakan yang “besar” pada algoritma Lucas-Kanade adalah dengan menggunakan *window* yang besar. Tetapi penggunaan window yang besar sering kali membuat gerakan yang ditangkap adalah gerakan yang tidak koheren. Algoritma *Pyramidal* Lucas-Kanade menyelesaikan permasalahan tersebut tanpa menghilangkan asumsi gerakan yang koheren.[4]

1.9 Open CV Library

Open CV adalah singkatan dari *Open Computer Vision*, yaitu suatu *library* gratis yang dikembangkan oleh *Intel Corporation* yang di khususkan untuk melakukan *image prosessing*. Tujuaannya adalah agar komputer mempunyai kemampuan yang mirip dengan cara pengolahan visual pada manusia. Open CV mempunyai API (*Aplication Programming Interface)* untuk *high level* maupun *low level*, terdapat fungsi-fungsi yang siap pakai, baik untuk *loading*, *saving*, akuisisi gambar maupun video.

Library ini terdiri dari fungsi-fungsi *Computer Vision* dan API (*Aplication Programming Interface*) untuk *image processing high level* maupun *low level* dan sebagai optimasi aplikasi *real-time*. OpenCV sangat disarankan untuk programmer yang akan berkutat pada bidang *Computer Vision*, karena *librar*y ini mampu menciptakan aplikasi yang handal, kuat dibidang *digital vision*, dan mempunyai kemampuan yang mirip dengan cara pengolahan visual pada manusia, karena *library* ini bersifat cuma-cuma dan sifatnya yang *open source*, maka dari itu OpenCV tidak dipesan khusus untuk pengguna arsitektur Intel, tetapi dapat dibangun pada hampir semua arsitektur. Saat ini para developer dari *Intel Corporation* telah membuat berbagai macam versi, yaitu:

1. OpenCV untuk bahasa pemograman C/C++
2. OpenCV untuk bahasa pemograman C# (masih dalam tahap pengembangan)
3. OpenCV untuk bahasa pemograman Java.

Untuk bahasa pemograman C# dan Java, karena masih dalam tahap pengembangan, maka kita membutuhkan *library* lain sebagai pelengkap kekurangan yang ada. Namun untuk bahasa pemograman C/C++ tidak memerlukan *library* lainnya untuk pemrosesan pada *computer vision*.[16]

1.9.1 Fitur Pada Open CV

Berikut ini adalah fitur2 pada *library* OpenCV:

1. Manipulasi data gambar (alokasi memori, melepaskan memori, kopi gambar, setting serta konversi gambar).
2. *Image*/Video I/O (Bisa menggunakan kamera yang sudah didukung oleh *library* ini).
3. Manipulasi matrix dan vektor serta terdapat juga routines linear algebra (*products*, *solvers*, *eigenvalues*, SVD).
4. Image processing dasar (*filtering*, *edge detection*, pendeteksian tepi, sampling dan interpolasi, konversi warna, operasi morfologi, histograms, *image pyramids*).
5. Analisis struktural.
6. Kalibrasi kamera.
7. Pendeteksian gerak.
8. Pengenalan objek.
9. Basic GUI (Display gambar/video, mouse/keyboard kontrol, scrollbar).
10. Image Labelling (line, conic, polygon, text drawing).

1.10 Emgu CV

OpenCV (*Open Source Comput er Vision)* adalah sebuah library fungsi pemrograman *real time* untuk *computer vision.* Emgu CV adalah wrapper .Net untuk OpenCV. Dengan EmguCV, fungsi-fungsi dalam OpenCV bisa dipanggil melalui bahasa pemrograman yang compatible dengan .NET seperti C#, VB, dan VC++. Selain itu, Emgu CV juga cross platform sehingga dapat di-compile lewat Mono dan dijalankan di atas sistem operasi Linux atau Mac OS.

Dari pengertian di atas telah diberikan deskripsi dari kedua *open source* tersebut. OpenCV merupakan *library* yang cukup terkenal di dunia *Computer Vision*. *Computer Vision* adalah salah satu bidang di teknologi informasi yang fokus pada pemrosesan *images* atau gambar yang diperoleh

darivdunia nyata untuk diekstrak dan diinterpretasikan informasinya. Untuk mempermudah *developer* dalam mengembangkan aplikasi yang menggunakan teknologi *computer vision*, digunakanlah *library* seperti VXL, Camellia, OpenCV, dan lainnya.

Maka dari itu EmguCV berperan untuk menjembatani C# dan OpenCV. EmguCV adalah wrapper .Net untuk OpenCV. Keuntungan menggunakan EmguCV yang paling utama adalah *library* ini sepenuhnya ditulis dengan bahasa pemrograman C# yang mana lebih aman karena pembuatan *object* atau pun *reference* di-manage oleh *garbage collector*. Ada dua konsep penting yang perlu diketahui terlebih dahulu sebelum menggunakan EmguCV. Pertama mengenai *layer* pada EmguCV. EmguCV terdiri dari 2 layer, yaitu *basic layer* dan *second layer*. *Basic layer* mengandung fungsi, struktur, dan enumerasi yang secara langsung merefleksikan apa yang ada di OpenCV. Dengan adanya layer inilah kita bisa memanggil fungsi-fungsi pada OpenCV dengan bahasa pemrograman C#. Sedangkan second layer mengandung kelas-kelas yang memanfaatkan keunggulan teknologi .NET.[19]