BAB II

LANDASAN TEORI

2.1 Citra

Secara harfiah citra atau *image* adalah gambar pada bidang dwimatra (dua dimensi). Ditinjau dari sudut pandang matematis, citra merupakan fungsi *continue* dari intensitas cahaya pada bidang dwimatra. Citra sebagai keluaran dari suatu sistem perekaman data dapat bersifat:

1. Optik berupa foto,
2. *Analog* berupa sinyal video seperti gambar pada *monitor* televisi,
3. *Digital* yang dapat langsung disimpan pada suatu pita magnetik.

Didalam bidang komputer, sebenarnya ada tiga bidang studi yang berkaitan dengan data citra, namun ketiganya berbeda.

2. 2 Pembangkitan Citra grafis

2.2.1 Sistem koordinat

Sistem koordinat yang selama ini dikenal antara lain adalah sistem koordinat kartesian yang digunakan untuk membedakan lokasi atau posisi sembarang titik atau obyek yang lain, sistem koordinat tersebut terdiri atas sistem koordinat kartesian dua dimensi dan tiga dimensi.

Lokasi setiap titik dalam sistem koordinat kartesian ditentukan oleh dua besaran. Nilai dua besaran tersebut apabila digambar akan membentuk suatu

sumbu koordinat mendatar dan tegak sumbu mendatar sering dikenal sebagai sumbu ***x*** dan sumbu koordinat tegak sebagai sumbu ***y***.

Sedangkan dalam sistem koordinat tiga dimensi terdapat 1 sumbu lain sebagai sumbu ***x*** dan sumbu ***y*** yang arahnya tegak lurus dengan sumbu ***y*** dan sekaligus tegak lurus dengan sumbu ***x*** sehingga lebih tepat dikatakan bahwa sumbu ini tegak lurus (menembus) bidang gambar dan disebut sebagai sumbu ***z***.

2.2.2 Pembangkitan *Pixel*

Proses pembangkitan *pixel* dapat dilakukan berdasarkan pola *digital* yang tersimpan dalam *memory digital*. Proses ini dapat dijelaskan secara sederhana dimana pada pengingat digital dan layar tampilan seolah-olah terdapat suatu piranti yang disebut sebagai *scanline*. *Scanline* ini akan “membaca” pola *digital* baris demi baris.

Setiapkali *scanline* ini “membaca” suatu baris pola *digital*, maka pengolahan tampilan menterjemahkan menjadi pola *pixel* yang terlihat dalam layar tampilan dengan demikian *scanline* pada layar tampilan bergerak sinkron terhadap scanline dalam *memory digital*.

2.2.3 Elemen gambar untuk menciptakan gambar pada komputer

Penghasilan citra grafis pada grafika komputer menggunakan primitif dasar. Primitif dasar ini memudahkan untuk merender (menggambar pada layar monitor) sebagaimana penggunaan persamaan geometri.

Primitif grafik dasar yang dipakai pada algoritma visual interaktif adalah.

1. Titik
2. Garis
3. Segi empat
4. Lingkaran

2.2.4 Titik

Titik pada komputer merupakan koordinat yang mengandung intensitas warna yang diletakan pada C*athoda-Ray-Tube* (CRT) monitor. Untuk meletakan titik pada koordinat ***x*** dan ***y*** (***x*** dan ***y*** merupakan besaran yang menyatakan pada posisi arah horizontal dan vertikal). Pada CRT monitor diperlukan program aplikasi untuk mengkonversi koordinat titik tersebut menjadi suatu nilai yang menyatakan lokasi dari peta layar monitor.

Peta pada layar monitor sebenarnya adalah linier, artinya layar monitor tidak mengenal sistem koordinat. Sebagai contoh jika lokasi-lokasi layar monitor (0,0) dan lokasi monitor (x_{max} , y_{max}) maka untuk mengkonversi koordinat suatu titik (***x,y***) menjadi suatu lokasi pada layar monitor digunakan rumus.

$$\text{Alamat } (x,y) = y\,(x_{max} + 1) + 1 \qquad \text{persamaan (2-1)}$$

Dimana alamat (***x,y***) merupakan lokasi sebenarnya pada layar monitor, ***x*** dan ***y*** adalah koordinat dari titik x_{max} adalah koordinat maksimum pada arah horizontal dan y_{max} adalah koordinat maksimum pada arah vertikal.

2.2.5 Garis

Suatu garis merupakan kumpulan titik yang letaknya berdampingan dan sangat dekat, sehingga yang terlihat hanyalah sepasang titik yang dapat disebut sebagai titik ujung dan titik pangkal garis tersebut. Jika hal ini kita lakukan pada layar tampilan maka seolah-olah kita menghidupkan *pixel* pada lokasi-lokasi tertentu sehingga yang kita lihat pada layar adalah suatu garis lurus.

Agar garis yang terlihat pada layar terlihat bagus maka harus memenuhi beberapa persyaratan tertentu, diantaranya garis harus terlihat lurus, garis harus berakhir pada titik yang tepat, garis harus mempunyai kerapatan *(density)* yang tetap untuk warna, garis harus mempunyai kerapatan atau tingkat kehitaman (*blackness)* yang tidak tergantung dari panjang dan sudut garis, dan garis harus dapat digambar dengan cepat.

Penggambaran garis dilakukan dengan menghitung posisi-posisi dimana sepanjang jalur lintasan antara dua posisi titik. Jika suatu titik (***x,y***) berada pada garis tersebut dinyatakan dengan koordinat ($\boldsymbol{x}_1, \boldsymbol{y}_1$) dan ($\boldsymbol{x}_2, \boldsymbol{y}_2$) maka persamaan untuk garis lurus adalah.

$$Y = m.x + b \qquad \text{persamaan (2-2)}$$

Dimana m mempersentasikan kemiringan garis yang dinyatakan dengan :

$$M = \frac{y_2 - y_1}{x_2 - x_1} \qquad \text{persamaan (2-3)}$$

Dan sebagai pemotong y yang dinyatakan dengan :

$$b - Y_1 - m.x \qquad \text{persamaan (2-4)}$$

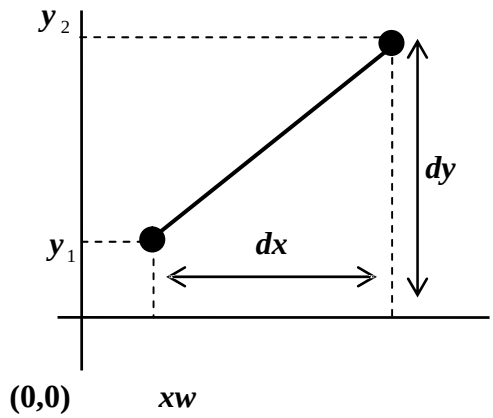
Untuk mencari interval ***y (dy)*** sepanjang garis harus dinyatakan dengan :

$$\Delta y = m.\Delta x \qquad \text{persamaan (2-5)}$$

Sedangkan untuk mencari interval ***x (dx)*** dinyatakan dengan :

$$\Delta x = \frac{\Delta y}{m} \qquad \text{persamaan (2-6)}$$

Untuk garis yang mempunyai gradien [**m**] < **1**, maka dx akan lebih kecil dari ***dy***, sedangkan garis yang mempunyai gradien [**m**]>**1** maka ***dx*** akan lebih besar dari ***dy***.



gambar 2.1 Garis yang menghubungkan titik (x_w, y_w) dan titik (x_v, y_v)

2.3 Grafika komputer (*computer graphics*)

Grafika komputer bertujuan menghasilkan citra (lebih tepat disebut grafik atau *picture*) dengan primitif-primitif geometri seperti garis, lingkaran, dan sebagainya. Primitif-primitif *geometri* tersebut memerlukan data *deskriptif* untuk melukis elemen-elemen gambar. Contoh data *deskriptif* adalah koordinat titik, panjang garis, jari-jari lingkaran, tebal garis dan sebagainya. Grafika komputer mempunyai peranan penting dalam *visualisasi* dan *virtual reality*.



Gambar 2.2 Alur dari Grafika komputer

2.3.1 Klasifikasi Penerapan Grafika Komputer

Sejumlah kriteria yang dapat dipakai untuk mengelompokan penerapan grafika komputer diantaranya adalah :

1. Tipe dari obyek dan gambar yang dihasilkan

   Beberapa contoh dari kriteria ini antara lain adalah gambar-gambar garis dari obyek 2D dan 3D (sering disebut *wired frame pictures*) misalnya garis-garis yang menunjukan struktur suatu bangunan, gambar garis untuk obyek untuk 3D dengan bagian tertentu yang tidak nampak untuk dihilangkan, gambar 2D dengan gray scale yang berbeda, gambar 3D yang

memperhatikan efek pencahayaan/bayangan, dan terapan-terapan lain yang sejenis.

2. Jenis interaksi dan derajat pengendalian objek/gambar dalam layer

   Dalam *offline plotting* gambar dapat dibentuk dari sekumpulan data yang diketahui atau diambil dari suatu basis data, sehingga gambar yang dihasilkan tidak dapat diubah kecuali dengan mengubah basis data yang diketahui. Sedangkan dalam *interactive plotting,* operator dapat berinteraksi dengan gambar sehingga operator dapat dapat berinteraksi dengan gambar sehingga operator dapat mengubah gambar yang tampil pada layer dengan menggunakan piranti yang tersedia. Dalam *interactive design,* operator dapat menggambar suatu bentuk tertentu sesuai kebutuhan mulai dari layer keadaaan kosong kemudian ditambahkan bentuk obyek yang dapat diubah dengan mudah oleh operator. Untuk dapat melihat bagian-bagian tertentu secara lebih mendetail dapat dilakukan dengan cara memperbesar atau memperkecil gambar tersebut.

3. Bagaimana gambar yang ditampilkan mempunyai arti bagi operator (*fule of the pictures)*

   Gambar yang dihasilkan dalam bidang-bidang tertentu seperti kartografi merupakan produk utama, sedangkan pada bidang-bidang lain gambar yang dihasilkan hanya merupakan visualisasi dari suatu obyek atau fenomena yang ingin diamati.

4. Hubungan antara obyek dan gambar tertampil

   Operator dapat menampilkan sebuah gambar dalam sekali tampil, sekelompok gambar yang saling berhubungan, ataupun dengan teknik lain

2.3.2 Kartografi

Dalam bidang kartografi, grafika komputer banyak digunakan untuk penyajian informasi geografis dan fenomena-feomena alamiah seperti untuk menyajikan peta-peta geografis, peta relief, peta eksplorasi minyak dan pertambangan, peta cuaca, peta kontur, peta kepadatan penduduk dan lain-lain.

2.3.3 *Computer aided design* (CAD)

Bidang ini memanfaatkan grafika computer sebagai alat Bantu untuk merancang komponen-komponen dari sistem-sistem mekanis, elektris, elektro mekanis, piranti-piranti elektronis dan piranti-piranti lainnya. Termasuk dalam kelompok ini adalah struktur. Seperti struktur bangunan, badan mobil, badan pesawat terbang, jaringan komputer dan sebagainya.

Tekanan utama dari pemakaian grafika computer dibidang ini adalah untuk memproduksi gambar dengan ketelitian yang sangat tinggi. Selain itu gambar yang diahsilkan CAP dapat dipakai sebagai masukan ke suatu sistem yang akan mengevaluasi model yang digambar sesuai kebutuhan dan dengan cara-cara tertentu operator dapat mengubah model yang sudah ada untuk disesuaikan dengan kebutuhan.

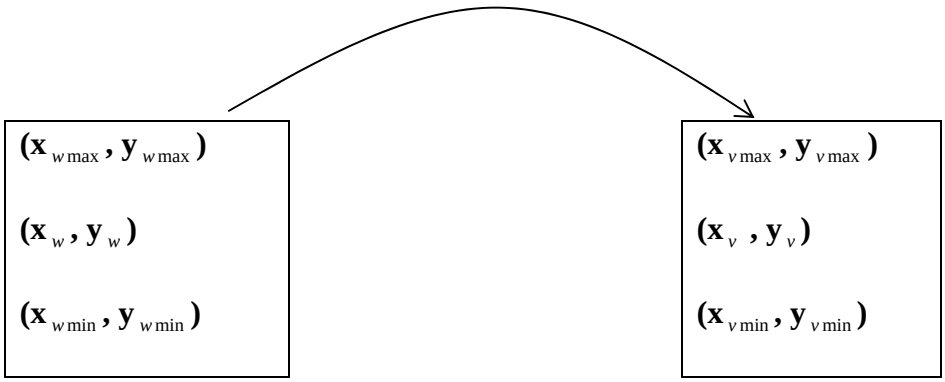
2.3.4 Antarmuka grafis (*Graphical User Interface*)

Dalam bidang ini grafika komputer seolah-olah menyediakan suatu bahasa antarmuka pemakai (*user interface)* yang baru, karena komunikasi antara user dengan komputer tidak lagi melalui suatu perintah yang harus diketikan dari papan ketik tetapi pemakai cukup memilih suatu simbol atau *icon* yang tersedia pada layer untuk mengerjakan suatu aktifitas yang akan dikerjakan pemilihan *icon* dapat dilakukan dengan beberapa piranti seperti *mouse, light pen, touch screen* dll.

2.3.5 Transformasi Dua Dimensi

Sistem koordinat komputer arah mendatar posistif adalah kekanan, jadi sesuai dengan koordinat kartesius. Tetapi arah gerak tegak positif adalah kebawah sehingga berlawanan dengan sistem koordinat kartesius. Karena kita sudah terbiasa dengan arah mendatar positif kekanan dan arah tegak positif keatas maka sistem koordinat akan mendatangkan kesukaran. Untuk menghindari kesukaran tersebut maka perlu dilakukan transformasi dari sistem koordinat layer ke sistem koordinat semesta yang mirip dengan sistem koordinat kartesius.

Pada gambar 2.2 melukiskan sebuah titik semesta yang akan dipetakan pada layer monitor dengan melalui proses transformasi. Misalkan koordinat titik tersebut si semesta adalah (x_w, y_w) dan dilayar monitor menjadi (x_v, y_v) tentu saja disemesta koordinat titik dinyatakan dalam sistem koordinat semesta dan dilayar monitor dinyatakan dalam sistem koordinat layer.



Gambar 2.3

Transformasi koordinat semesta ke koordinat layer.

Dengan menggunakan persamaan.

$$\frac{xv - xv_{\min}}{xv_{\max} - xv_{\min}} = \frac{xw - xw_{\min}}{xw_{\max} - xw_{\min}}$$

$$\frac{yv - yv_{\min}}{yv_{\max} - yv_{\min}} = \frac{yw - yw_{\min}}{yw_{\max} - yw_{\min}}$$

Hubungan kedua koordinat tadi adalah

$xv = xv_{\min} + (xw - xw_{\min}) sx$

$yv_{\max} - yu_{\min} + (yw - yw_{\min}) sy$ persamaan (2-7)

Dengan A,B,C,D adalah tetapan transformasi yang akan dicari dari absis titik sudut kiri dan titik sudut kanan diperoleh hubungan

$$x_{v\min} = A.\,X_{w\min} + B \qquad\qquad X_{v\min} = C.Y_{\max} + D$$

Sehingga diperoleh

$$A = \frac{xv_{\max} - xv_{\min}}{xw_{\max} - xw_{\min}} \qquad \text{persamaan (2-8)}$$

Dari koordinat kedua sudut tadi diperoleh hubungan.

$$Y_{v\min} = Ax.\,x_{w\min} + D \qquad\qquad Y_{v\ \max} = Cx.\ X_{w\max} + A$$

Maka

$$C = \frac{yv_{\max} - yv_{\min}}{yw_{\max} - yw_{\min}} \qquad \text{persamaan (2-9)}$$

$$D = Y_{v\min} - Y_{w\min}.A$$

Dengan diketahui tetapan transformasi tersebut maka persamaan 2-7 dapat dihitung dari persamaan 2-8 dan persamaan 2-9. tampak bahwa selain batas *x* semesta diperlukan juga batas-batas bidang gambar pada layar monitor $(\boldsymbol{x}_{1\min}, y_{1\min})$ dan $(\boldsymbol{x}_{1\max}, y_{1\max})$.

Transformasi matrik adalah angka-angka yang dimodifikasi oleh suatu operasi matematika. Transformasi geometri adalah perubahan posisi suatu objek, baik posisi itu terjadi karena penggesaran (translasi) perputaran (rotasi) atau pengskalaan (scaling).

2.3.6 Translasi

Translasi digunakan pada suatu objek untuk meletakan kembali objek tersebut sepanjang jalur lurus dari suatu lokasi koordinat ke lokasi koordinat yang lain. Untuk mentranslasi suatu titik (***x,y***) dengan penggesaran ($t_{x,}t_y$) menjadi titik (x`,y`) adalah

$$X^{|} = x + t_x \qquad Y^{|} = y + t_y \qquad \text{persamaan (2-10)}$$

Dimana x,y adalah koordinat asal dari titik, t_x dan t_y adalah pergesaran pada sumbu x dan sumbu y,x` dan y` adalah koordinat titik sesudah translasi.

Secara singkat persamaan 2-10 dapat ditulis kedalam bentuk matriks dua dimensi. Jika P mempersantasikan koordinat (*x*,*y*), P` mempersentasikan koordinat (*x*`, *y*`) dan T mempersentasikan pergeseran t_x dan t_y. Maka persamaan 2-10 dapat ditulis kedalam bentuk matriks menjadi :

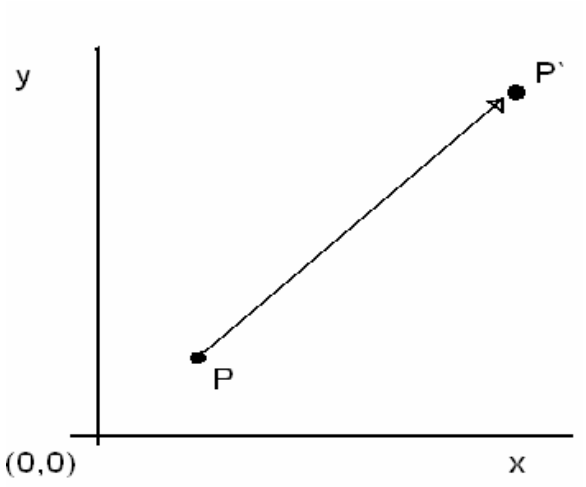
$$\text{P} = \begin{bmatrix} x \\ y \end{bmatrix} \qquad \text{P`} = \begin{bmatrix} x^{|} \\ y^{|} \end{bmatrix} \qquad \text{T} = \begin{bmatrix} t_x \\ t_y \end{bmatrix} \quad \text{persamaan (2-11)}$$

Dimana untuk mentranslasi suatu titik P menjadi P` adalah :

P` = P + T persamaan (2-12)

Atau

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} + \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$ persamaan (2-13)



gambar 2.4 Translasi titik dari posisi P ke posisi P` dengan pergeseran sebesar T.

2.4 Algoritma *Z-Buffer*

Tampilan yang berupa *ruster* gambar terdiri dari *frame* yang dapat ditampilkan pada permukaan atau *buffer* garis-garis *polygon* atau garis-garis lainnya. Apabila kita menggunakan tampilan tersebut, cara lain adalah dengan menyembunyikan bagian permukaan sebagai *background* untuk disusun pada *frame* tersebut sehingga kita bisa melihat tampilan pada warna-warna *pixel* tersebut berupa bagian dari permukaan yang tertutup oleh bagian lainnya. Untuk melakukan hal tersebut kita harus melakukan semua proses pada permukaan tersebut untuk dijadikan suatu kumpulan proyek pada bagian yang dapat kita lihat setelah adanya penambahan objek baru. Kita harus mengurutkan *polygon* atau garis-garis tersebut secara berurutan. Sesuai dengan posisi dari area yang telah ada. Langkah ini berupa pengurutan bidang geometri atau pengkotakan pada area permukaan, area tersebut harus memiliki titik koordinat tengah yang tersembunyi pada permukaan tersebut.

Salah satu permasalahan yang sering timbul pada algoritma *Z-Buffer* ini adalah ketika kita membuat suatu area yang sangat luas. Algoritma *Z-Buffer* digunakan untuk menentukan garis koordinat z. Garis-garis koordinat tersebut membantu pada saat kita mengurutkan garis-garis atau *polygon* dengan teks – teks posisi pada permukaan yang akan ditampilkan. Ketika frame-frame tersebut dihilangkan atau di *clear* maka secara otomatis algoritma ini akan memiliki nilai negatif.

Penginilisasian nilai mungkin ada pada saat penentuan posisi pada *background*. Garis-garis polygon tersebut akan dimasukan 1 demi 1 ke *frame*

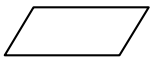
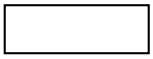
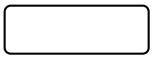
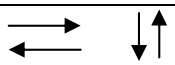
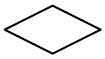
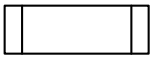
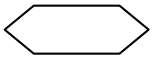
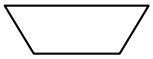
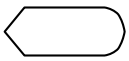
tersebut atau menampilkan *file* yang telah ada dengan menggunakan algoritma kompresi. Jika permukaan baru berada pada bagian belakang dari objek yang baru maka objek baru tersebut akan berada diatas.

Jika kita akan memodifikasi sistem dengan algoritma *Z-Buffer* maka jika ada objek dipermukaan tersebut akan dihilangkan objek lama dan digantikan obyek baru. Semua yang kita butuhkan adalah ketika kita akan mulai dan memasukkan nilai dan ketika kita akan merubah nilai-nilai tersebut setiap saat dan secara otomatis koordinat y untuk bagian pinggir garis akan muncul. Algoritma *Z-Buffer* tersebut akan memakai bagian dari memori untuk digunakan ketika akan memasukkan objek atau memodifikasinya, dengan demikian kita harus menyediakan ruang pada memori untuk digunakan ketika akan membuat atau memodifikasi objek pada area tersebut. Bagaimanapun juga algoritma ini dapat dibuat secara sederhana. Sebagai masalah kecepatan ketika proses dilakukan. Metode ini sangat terkenal sekali untuk mengatasi masalah menyembunyikan area.

2.5 *Flowchart*

Flowchart adalah blok-blok diagram yang menggambarkan alur sebuah program dan merupakan alat bantu yang digunakan untuk menggambarkan sistem picikan yang tersusun secara baik GEO[5]. Saat ini telah diterbitkan standar untuk diagram alir oleh Institut Nasional Amerika (ANSI). Gambar simbol-simbol *flowchart* dalam standar ANSI dapat dilihat pada tabel 2.1

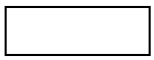
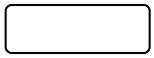
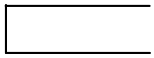
Tabel 2.1 Simbol-simbol *Flowchart* dalam standar ANSI

Simbol	**Keterangan**
	Input/ Output
	Proses/ Pengolahan
	Penghubung
	Akhir, Penyela
	Garis Alir
	Keputusan
	Proses yang dirumuskan
	Persiapan
	Operasi Manual
	Pita Magnetic
	Dokumen
	Tampilan

2.5.1 Diagram Arus Data

Diagram arus data merupakan dokumen arus data yang logis dalam suatu sistem. Diagram ini biasanya digunakan dalam sistem pemrosesan transaksi dan untuk menggambarkan aliran data dan fungsi-fungsi yang terjadi dalam suatu sistem. Diagram arus data menyatakan sumber dan tujuan data, arus data, langkah pemrosesan dan penyimpanan data. Dan simbol diagram arus berdasarkan notasi yourdan.

Tabel 2.2 Simbol Diagram Alir Data

Simbol	Keterangan
	Asal atau tujuan data
	Aliran data
	Proses yang mengubah aliran data
	Penyimpanan data

2.6 Visual Basic 6

2.6.1 Sejarah Visual Basic

Visual Basic adalah salah satu bahasa pemrograman komputer. Bahasa pemrograman adalah perintah-perintah yang dimengerti oleh komputer untuk melakukan tugas-tugas tertentu. Bahasa pemrograman Visual Basic, yang dikembangkan oleh Microsoft sejak tahun 1991, merupakan pengembangan dari pendahulunya yaitu bahasa pemrograman BASIC (*Beginner's All-purpose Symbolic Instruction Code*) yang dikembangkan pada era 1950-an. Visual Basic merupakan salah satu *Development Tool* yaitu alat bantu untuk membuat berbagai macam program komputer, khususnya yang menggunakan sistem operasi Windows. Visual Basic merupakan salah satu bahasa pemrograman komputer yang mendukung objek (*Object Oriented Programming* = OOP).

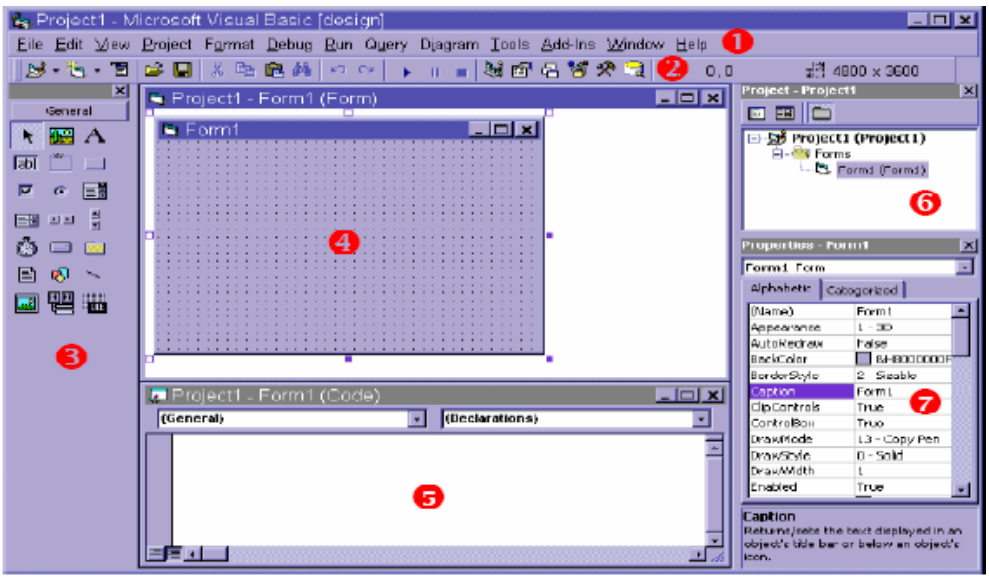
2.6.2 Integrated Development Environment (IDE) VB 6

Merupakan lingkungan pengembangan dari Visual Basic 6 yang terintegrasi. Ketika pertama kali Visual Basic diaktifkan maka akan muncul muncul tampilan seperti pada gambar 2.5.



gambar 2.5 Layar *New Project*

Setelah kita keluar dari layar *new project*, selanjutnya akan tampil layar kerja atau area kerja dari Visual Basic 6.



gambar 2.6 Layar kerja Visual Basic 6

Keterangan dari gambar diatas :

1. Menubar

   Menubar berfungsi untuk memilih tugas-tugas tertentu, seperti memulai, membuka dan menyimpan *project*, mengompilasi *project* menjadi file *executable* (*exe*) dan lain-lain.

2. Toolbar

   Toolbar memiliki fungsi yang sama seperti menubar, dan juga berfungsi seperti jalan pintas, Karena lebih praktis dalam penggunaannya.

3. Toolbox

   Toolbox berisi komponen-komponen yang merupakan sarana untuk membentuk *user interface*.

   Bila *Toolbox* tidak muncul klik tombol *Toolbox* () pada bagian *Toolbar* atau klik menu *View* > *Toolbox*.

4. Jendela Form Designer

   Form designer merupakan tempat untuk merancang *user interface*, dimana kontrol-kontrol yang dibutuhkan bisa diletakan.

   Bila Jendela *Form designer* tidak muncul klik tombol *View Object* () pada bagian *Project Explorer* atau klik menu *View* > *Object*.

5. Jendela Kode

   Jendela kode merupakan tempat untuk menuliskan kode-kode bagi objek yang dibuat. Melalui jendela ini dapat mengatur karakteristik dari form yang dibuat.

   Bila Jendela *Kode* tidak muncul klik tombol *View Code* ( ) di pada bagian *Project Explorer* atau klik menu View > *Code*.

6. Jendela Project

Jendela *project* adalah jendela yang menampilkan semua file yang berhubungan dengan aplikasi atau *project* yang saat itu sedang dijalankan.

Bila *Project Explorer* tidak muncul klik tombol *Project Explorer* () pada bagian *Toolbar* atau klik menu *View* > *Project Explorer*.

7. Jendela Properties

Jendela properties ini berisi daftar property untuk objek (form atau control) yang dipilih dan berfungsi untuk mengatur karakteristik, seperti warna, ukuran dan lain-lain.

Bila Jendela *Properties* tidak muncul klik tombol *Properties Window* () pada bagian *Toolbar.*

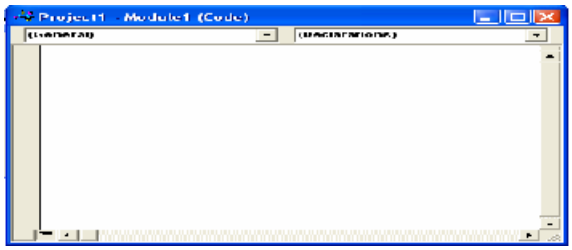
Banyak Sekali *tool–tool* dan jendela- jendela pendukung lainnya yang ada di lingkungan visual basic 6 sebagai alat untuk mengembangkan aplikasi yang akan dibuat. Seperti jendela *form layout* yang menunjukan tampilan form pada saat dijalankan.



gambar 2.7 Tampilan Jendela Form Layout

Ada juga jendela modul yang merupakan tempat untuk menuliskan kode-kode yang lebih bersifat global. Pada jendela modul ini, bisa mendeklarasikan variabel dan fungsi dengan awalan *private* maupun *public*. Jika diberi awalan *private*, variabel atau fungsi tersebut hanya berlaku untuk pemakaian modul yang

bersangkutan. Sedangkan awalan *public* akan mengakibatkan variabel atau fungsi tersebut bersifat global. Jendela modul ini bisa ditampilkan dengan mengklik menu *project* pada menubar, kemudian pilih *add module*.



gambar 2.8 Tampilan Jendela Modul

2.6.3 Struktur program Visual Basic 6

Pada saat membangun sebuah aplikasi, digunakan *project* untuk mengatur semua file yang digunakan dalam suatu aplikasi.

Project itu sendiri terdiri atas :

a. Sebuah file project yang mempertahankan semua komponen (.vbp)
b. Sebuah file untuk formnya (.frm)
c. Sebuah file untuk setiap module standard **(.bas)**
d. Sebuah atau beberapa file yang berisi kontrol Active – X **(.ocx)**
e. Sebuah file resource **(.res)**, file ini tidak harus ada.

File Project adalah semua daftar file dan objek yang telah dihubungkan dengan project, termasuk informasi pengaturan yang ada.

This document was created with Win2PDF available at http://www.daneprairie.com.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.