BAB II

LANDASAN TEORI

2.1. Pengertian Grafika Komputer

Apakah grafika komputer itu? Hill (1990:2) menyatakan bahwa grafika komputer merupakan sekumpulan alat yang digunakan untuk membuat gambar (*to create picture*) dan berinteraksi dengan gambar dengan cara-cara seperti yang biasa digunakan (*and to interact with them in natural ways*). Peralatan tersebut dapat berupa perangkat keras (*hardware*) ataupun piranti lunak (*software*) semua peralatan tersebut memberikan fasilitas kepada programmer untuk membuat program-program grafis yang baik. Secara umum program-program komputer grafis ditujukan untuk memudahkan interaksi manusia dengan komputer. Dengan memvisualisasikan kata-kata menjadi gambar maka manusia akan lebih mudah berinteraksi dengan komputer, perintah-perintah tidak lagi diberikan dengan cara menulis tetapi dengan cara menunjuk area atau gambar tertentu.

Secara umum grafika komputer adalah suatu desain gambar baik itu berupa garis, bidang ataupun ruang yang mengacu pada koordinat *cartesian*. Koordinat *cartesian* grafik dua dimensi mempunyai dua sumbu yaitu sumbu X dan Y. Sumbu X mempunyai arah ke kanan untuk nilai positif dan kekiri untuk nilai negatif. Sedangkan sumbu Y mempunyai arah ke atas untuk nilai positf dan ke bawah untuk nilai negatif. Sedangkan koordinat cartesian untuk grafika tiga dimensi memiliki tiga sumbu yaitu X, Y dan Z.

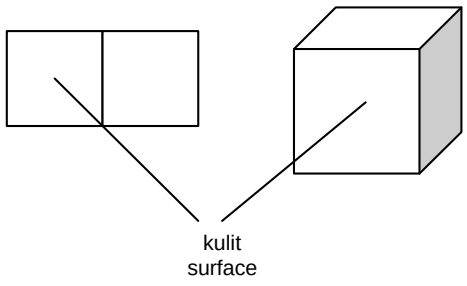
Komputer telah dirancang sebagai media yang mempunyai kemampuan menggambar grafik dengan cepat dan efisien. Pembuatan visual grafik tiga dimensi pada dasarnya memberikan perubahan-prubahan pada objek diantaranya translasi, perputaran, perubahan skala atau ukuran dan pantulan bayangan *(reflection*).

Sebelum tampilan tiga dimensi yang telah diletakan atau disesuaikan didalam titik koordinat harus ditentukan terlebih dahulu koordinat titik pandang. Koordinat titik pandang berpengaruh pada orientasi dan posisi objek yang akan diproyeksikan kedalam sebuah piranti tampilan yang terpilih.

2.2. Benda Tiga Dimensi

2.2.1. Anatomi benda tiga dimensi

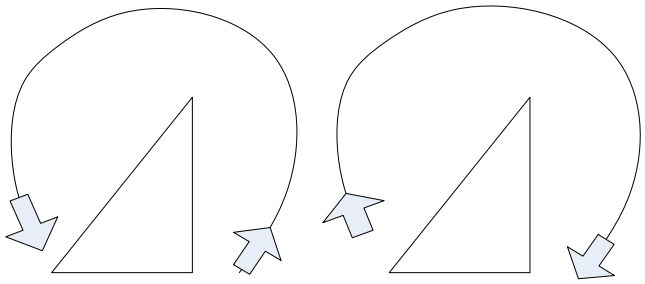
Benda tiga dimensi disusun dari sekumpulan kulit (*surface*). Kulit dapat dibuat dari rangkaian *polygon*. Gambar 2.1 memberikan contoh bagaimana kulit digunakan untuk menyusun benda tiga dimensi.



Gambar 2.1 kulit penyusun benda tiga dimensi

Bentuk polygon yang paling tepat digunakan untuk membuat permukaan benda tiga dimensi adalah *polygon* segitiga *(triangle).* Hal ini disebabkan *polygon* segitiga akan selalu berada dalam keadaan *planar* (datar) sementara *polygon*

dengan verteks lebih dari 3 dapat berada dalam kondisi *non-planar* karena salah satu verteks tidak berada di lokasi yang sejajar. Satu hal yang harus diperhatikan pada saat menggunakan *polygon* segitiga adalah *winding*. *Winding* adalah urutan dan arah verteks-verteks penyusun *polygon. Winding* menganggap bahwa arah verteks berlawanan jarum jam sebagai tampak depan sedangkan arah verteks searah jarum jam dianggap sebagai bagian belakang benda.



Gambar 2.2 Arah verteks : (a) berlawanan jarum jam; (b) searah jarum jam

Pada gambar 2.2.a *polygon* menggunakan winding berlawanan jarum jam karena didefinisikan sebagai urutan verteks v3-v4-v5 sedangkan pada gambar 2.2.b menggunakan *winding* searah jarum jam karena urutan verteks yang digunakan v3-v5-v4.

2.2.2. Strktur Data Benda 3 Dimensi

Sebuah benda tiga dimensi dapat disusun dengan menghubungkan sejumlah titik. Kumpulan titik-titik yang saling terhubung tersebut disebut sebagai *wireframe* atau *kerangka*, sedangkan benda yang disusun dari kumpulan titik dan permukaan tersebut disebut sebagai *mesh*. Perhatikan Gambar 2.3, kubus yang ada

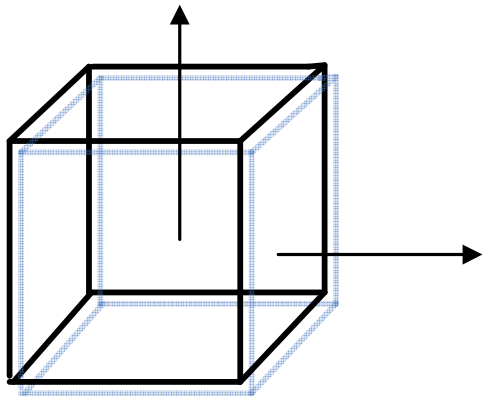
v4

v5 v3

V3-v4-v5

(a)

di gambar tersebut dapat disusun berdasarkan data yang ada pada Tabel 2.1 dan tabel 2.2



Gambar 2.3 Kubus

Tabel 2.1verteks penyusun kubus

Verteks	X	Y	Z
0	-1	-1	+1
1	+1	-1	+1
2	+1	+!	+1
3	-1	+1	+1
4	+1	-1	-1
5	-1	-1	-1
6	-1	+1	-1
7	+1	+1	-1

Tabel 2.2 Permukaan *(face)* kubus

Face	V1	V2	V3	keterangan
F1.a	0	1	2	Depan
F1.b	0	2	3	Depan
F2.a	1	4	7	Kanan
F2.b	1	7	2	Kanan
F3.a	4	5	6	Belakang
F3.b	4	6	7	Belakang
F4.a	5	3	0	Kiri
F4.b	5	6	3	Kiri

F6.a	3	2	7	Atas
F6.b	3	7	6	Atas
F6.a	0	1	4	Bawah
F6.b	0	4	5	bawah

Tabel 2.2 Menyatakan urutan verteks yang menyusun sebuah permukaan. Untuk setiap permukaan didefinisikan sebagai *polygon* segitiga (*triangle*), sebagai contoh permukaan kubus bagian depan didefinisikan melalui dua buah segitiga, yaitu F1.a dan F1.b yang masing-masing disusun dari verteks 0, verteks 1, dan verteks 2, serta verteks 0, vertek 2 dan verteks 3.

2.3 Teknik Pencahayaan

Tampilan yang terlihat nyata pada sebuah *scene* diperoleh dari hasil memanipulasi objek dengan cara proyeksi-proyeksi perspektif dan dengan memberikan sedikit efek pencahayaan yang alami pada permukaan objek yang terlihat. Sehingga objek tiga dimensi yang telah atau mendekati selesai belum sempurna tanpa adanya efek pencahayaan. *Illumination mode* disebut juga *light intensity* atau kadang-kadang dikenal sebagai *shading model* yang digunakan untuk mengkalkulasi atau menghitung intensitas cahaya yang terlihat di sebuah titik permukaan objek.

Pewarnaan dan pemberian efek cahaya yang kita lihat pada suatu objek merupakan proses yang umum dilakukan dengan menyertakan prinsip-prinsip dasar ilmu fisika. Pada dasarnya, efek pencahayaan yang diuraikan dengan mengacu pada interaksi antar tenaga elektromagnetis dengan permukaan objek.

Teknik pencahayaan dengan komputer grafik sering mengacu pada hukum-hukum fisika yang berhubungan dengan intensitas cahaya permukaan. Untuk menyederhanakan perhitungan sebuah intensitas cahaya, dapat menggunakan perhitungan empiris yang didasarkan pada kalkulasi *photometric* yang disederhanakan. Contoh yang lebih akurat, seperti *radiosity algoritma* dimana perhitungan intensitas cahaya dengan mempertimbangkan jarak permukaan objek dengan sinar cahaya di dalam sebuah *scene.*

2.3.1 Pantulan Cahaya

Ketika terlihat suatu objek yang bercahaya sedangkan objek tersbut bukan sebuah sumber cahaya, itu berarti cahaya terlihat pada permukaan objek tersebut adalah cahaya pantulan.Total cahaya yang di pantulkan adalah penjumlahan dari kontribusi sember cahaya dan pemantulan cahaya di permukaan objek lain di dalam *scene*, dalam arti suatu permukaan objek yang tidak bisa secara langsung di sorot atau diberikan penerangan oleh sumber cahaya (*light-emiting source*) akan tetap terlihat jika objek yang letaknya berdekatan dapat penerangan.

Sebuah objek yang mempunyai permukaan yang kasar maka cahaya yang di pantulkannya cenderung akan menyebar ke segala arah, cahaya yang tersebar ini di sebut *diffuse reflction.* Sedangkan apabila sumber cahaya menciptakan cahaya yang terang atau sebuah titik terang di sebut *specular reflection.* Efek dari cahaya terang ini lebih ditekankan pada permukaan yang berkilau daripada permukaan yang tumpul. Besarnya sudut yang dihasilkan oleh *spekular reflection* terhadap

vektor normal permukaan sama besar dengan sudut di mana cahaya masuk terhadap vektor normal permukaan tetapi arahnya berlawanan.

Selain objek yang memperhatikan data pantul cahaya yang sempurna (*specular replections*) di atas mempunyai suatu cakupan yang terbatas dari posisi sudut pandang di sekitar vektor R (*speculer rflection*). Sedangkan sebuah permukaan yang tumpul mempunyai cakupan yang lebih luas, yang kemudian di kembangan oleh Phong Bui Toung dan dikenal Phong ***Specular-Reflection.model*** atau hal yang sederhana dari Phong model yaitu dengan menetapkan *Specular reflection* sebanding dengan cos ¢ . sudut ¢ ditetapkan antara 0 ^{o} sampai 90 ^{o}, sedemikian sehingga cos ¢ mempunyai variasi nilai antara 0 sampai 1.

Dalam sebuah sumber cahaya tunggal, kita dapat mengkombinasikan antara *diffuse* dan *specular reflection* pada suatu titik diatas sebuah permukaan yang diberi pencahayaan sebagai berikut :

$$I : I_{diff} + I_{spec}$$

$$I = Ka.Ia + Kd.Ie\ (N \circ L) + Ks.Ie\ (N \circ H) \qquad (2.1)$$

Menggambarkan efek pencahayaan terhadap sebuah permukaan yang diproduksi oleh berbagai faktor. Apabila ditempatkan lebih dari satu sumber cahaya di dalam *scene*, maka akan diperoleh pantulan cahaya pada titik permukaan dengan menjumlahkan dari sumber tersebut seperti :

$$I = Ka.Ia + \sum_{i=1}^{n} I\,li\,[\,Kd\,(N \circ Li) + Ks\,(N \circ H)^{ns}\,] \qquad (2.2)$$

Suatu permukaan yang transparan secara umum akan menghasilkan dua pengaruh yaitu memantulkan dan meneruskan (menyerap) cahaya. Kontribusi

relatif dari cahaya yang dipancarkan tergantung pada tingkatan ketransparanan permukaan objek itu sendiri.

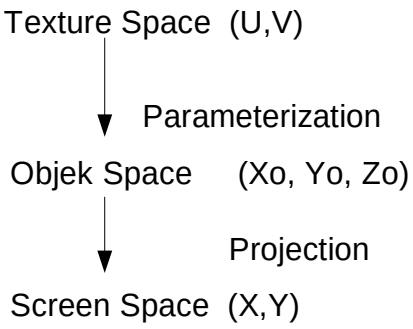
Ketika permukaan yang transparan diberikan pencahayaan, maka persamaan intensitas harus dirubah supaya meliputi kontribusi dari cahaya yang melintas permukaan itu. Dalam banyak kasus, cahaya yang dipancarkan akan diturunkan dan dipantulkan dibelakang permukaan objek.

2.4 Algoritma texture mapping

Texture mapping dapat dikatakan sebagai primitif grafika komputer seperti halnya titik dan garis. Aplikasi *texture mapping* tidak dapat dipisahkan dalam pemodelan tiga dimensi karena *texture mapping* selalu dipakai dalam semua pemodelan tiga dimensi. *Texture mapping* adalah teknik *shading* untuk pengolahan gambar yang memetakan sebuah fungsi pada permukaan tiga dimensi dalam *scene.*

Fungsi yang dipetakan mencakup satu dimensi, dua dimensi, dan tiga dimensi dan dapat digambarkan sebagai *array* atau fungsi matematika atau gambar. Sebagai contoh , tekstur satu dimensi diwakili oleh gambar lapisan, batuan, tekstur dua dimensi diwakili oleh gelombang atau permukaan yang bergelombang, dan tekstur tiga dimensi diwakili oleh awan, kayu atau marmer. Namun, dalam tugas akhir ini kita memakai tekstur dua dimensi yang akan diubah menjadi tekstur tiga dimensi. *Texture mapping* dapat digunakan untuk menggambarkan banyak parameter permukaan disamping warna seperti *bump mapping* untuk memperjelas karakteristik permukaan yang bergelombang,

Transparency mapping untuk mengatur intensitas cahaya permukaan tembus pandang, *Specularity mapping* untuk mengubah kehalusan permukaan dan *Illumination maaping* untuk memodelkan distribusi cahaya yang datang dari berbagai arah. Namun dari sumua itu yang paling penting adalah *Geometrical mapping, geometrical mapping* secara keseluruhan ditentukan dengan dengan transformasi tiga dimensi terhadap kamera, tansformasi model yang menggambarkan geometri *scene* dan pemberian parameter pada permukaan dengan tujuan memetakan tekstur ke permukaan, ada 2 tahap yang harus dilakukan untuk memetakan teksture dari *space* ke *screen space* seperti pada gambar dibawah ini.



Gambar 2.4 Tahap *texture mapping*

Sistem koordinat pada masing-masing *space* adalah pada *texture space koordinat* yang dipakai adalah (U,V), objek *space* (*scene*) memakai koordinat (Xo, Yo, Zo) dan *screen space* memakai koordinat (X,Y)

2.4.1 Tahap *texture maaping*

Berikut ini penjelasan dua tahap pemetaan tekstur dua dimensi ke permukaan tiga dimensi.

2.4.1.1 Parameterisasi

Memetakan tekstur dua dimensi ke permukaan tiga dimensi memerlukan pemberian parameter pada permukaan. Parameter yang dimaksud dapat berupa koordinat U dan V, seperti layaknya operasi *texture mapping*, atau vektor normal atau cahaya, seperti dalam *illumination mapping* atau dengan koordinat Xo, Yo, dan Zo untuk objek zat padat yang tampak terpotong.

Sebagai contoh parameterisasi segitiga dapat dilakukan dengan menentukan koordinat texture (U,V) pada tiga titiknya. Untuk bentuk yang lebih rumitnya dari segitiga, misalnya persegi empat. Parameterisasinya dilakukan dengan membagi bentuk tersebut menjadi segi tiga.

2.4.2 Teknik *texture mapping*

Sebenarnya ada banyak teknik *texture mapping* yang dapat digunakan untuk proses rendering. Pemilihan teknik ini berdasarkan pertimbangan bahwa teknik ini dapat mewakili teknik *texture mapping* yang lain. *Perfect mapping* mewakili teknik *texture mapping* yang hasilnya memiliki kualitas tertinggi tetapi memerlukan waktu rendering yang lebih lama.

2.4.2.1 Perfect mapping

Perfect mapping tergolong dalam teknik *texture mapping* yang sederhana karena dalam *perfect mapping* hanya dihitung lebar dan tinggi objek tiga dimensi dan menentukan lebar tekstur yang akan dipetakan sesuai dengan ukuran objek tiga dimensi tersebut. *Perfect mapping* ini disebut juga dengan *true mapping*

karena pada proses penempatan *pixel texture* pada permukaan dihitung satu persatu berdasarkan atribut yang telah diperoleh dari objek tiga dimensinya.

Hasil algoritma *perfect mapping* ini adalah permukaan objek tiga dimensi ditutupi oleh tekstur yang panjang dan lebar teksturnya telah disesuaikan dengan panjang dan lebar objek tiga dimensinya (tekstur di-*strech* untuk menutupi permukaan objek tiga dimensi). Parameterisasi algoritma *perfect mapping* ini adalah sebagai berikut :

$$U = \frac{(Q - loc).V1}{V1.V2}$$

$$V = \frac{(Q - loc).Va}{Va.Va} \qquad (2.3)$$

Dengan :

U = posisi pixel pada tekstur arah mendatar (u)

V = posisi pixel pada tekstur arah tegak (v)

Q = lokasi pada objek tiga dimensi (X, Y, Z)

Loc = lokasi objek tiga dimensi (X, Y, Z)

V1 = panjang objek tiga dimensi (X, Y, Z)

V2 = lebar objek tiga dimensi (X, Y, Z)

2.5 Transformasi

Transformasi pada 3 dimensi pada dasarnya sama dengan transformasi pada 2 dimensi, hanya pada 3 dimensi kita juga memperhitungkan sumbu z. sama seperti pada 2 dimensi, ada 3 transformasi dasar yang dilakukan terhadap verteks, yaitu:

1. translasi
2. penskalaan
3. Rotasi

Titik hasil transformasi dapat diperoleh melalui rumus 2.4 rumus 2.4 disebut sebagai *Affine Transformation*.

$$Q = P * M + tr \qquad (2.4)$$

Dimana :

Q = (Qx,Qy,Qz) menyatakan matriks 1x3 yang berisi titik hasil transformasi.

P = (Px, Py, Pz) menyatakan matrik 1x3 yang berisi titik yang akan ditransformasi.

Tr = (trx, try, trz) menyatakan matrik 1x3 yang berisi banyaknya pergeseran pada sumbu x, y dan z.

M = Matriks mentransformasi berukuran 3x3 seperti pada 2.5

$$\begin{bmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ m_{20} & m_{21} & m_{22} \end{bmatrix} \qquad (2.5)$$

2.5.1 Translasi

Transformasi translasi merupakan operasi yang menyebabkan perpindahan objek dua dimensi dari satu tempat ke tempat yang lainnya. Perubahan ini berlaku dalam arah yang sejajar dengan sumbu x , sumbu y dan sumbu z. Dalam operasi translasi, setiap titik pada suatu entitas yang ditranslasi bergerak dalam jarak yang

sama. Pergerakan tersebut dapat berlaku dalam arah sumbu x , y dan saja atau ketiganya.

Untuk mentranslasikan suatu titik (x,y,z) dengan pergeseran sebesar tx, ty dan tz menjadi titik (x', y', z') adalah:

$$X' = x + tx$$

$$Y' = y + ty$$

$$Z = z + tz \qquad (2.6)$$

Dimana x,ydan z adalah koordinat asal dari titik. tx adalah pergeseran pada sumbu x, ty adalah pergeseran pada sumbu y, tz adalah pergeseran pada sumbu z sedangkan x', y' dan z' adalah koordinat titik sesudah ditranslasikan.

Secara singkat persamaan diatas dapat ditulis kedalam bentuk matriks tiga dimensi. Jika P mepresentasikan koordinat (x, y dan z), P' mempresentasikan koordinat (x',y' dan z') dan T mempresentasikan pergeseran tx, tydan tz, maka persamaan di atas dapat ditulis kedalam bentuk matrik menjadi:

$$P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad P' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} \quad T = \begin{bmatrix} tx \\ ty \\ tz \\ 1 \end{bmatrix} \qquad (2.7)$$

Dimana untuk mentranslasikan suatu titik p menjadi p' adalah

$$P' = P * T$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

(2.8)

Untuk memudahkan perhitungan maka kita dapat menggnakan bentuk homogeneus transformation, yaitu dengan mengubah matriks transformasi menjadi berukuran 4x4 seperti pada rumus(2.9)

$$\begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \\ tr_x & tr_y & tr_z & 1 \end{bmatrix} \qquad (2.9)$$

2.5.2 Penskalaan

Penskalaan dilakukan dengan mengisi tr = (0, 0, 0) dan matrik M diatur seperti 2.10

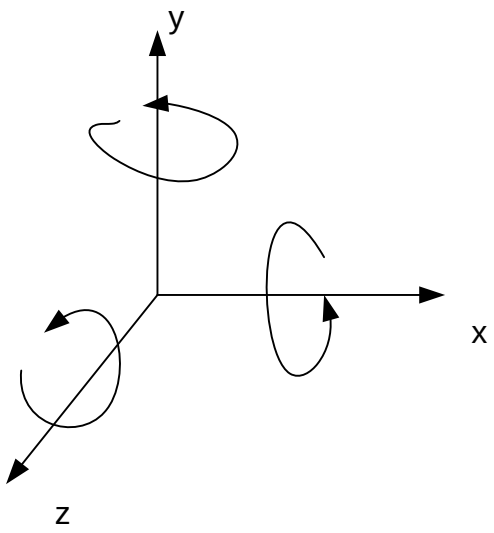
$$\begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.10)$$

2.5.3 Rotasi

Berbeda dengan rotasi 2 dimensi yang menggunakan titik pusat (0,0)sebagai pusat perputaran, rotasi 3 dimensi menggunakan sumbu koordinat sebagai pusat perputaran. Dengan demikian ada 3 macam rotasi yang dapat dilakukan, yaitu:

1. Rotasi sumbu x
2. Rotasi sumbu y
3. Rotasi sumbu z

Gambar 2.6 memperlihatkan bagaimana hubungan antara rotasi 3 dimensi dan sumbu rotasi.



Gambar 2.6 Rotasi dan sumbu rotasi.

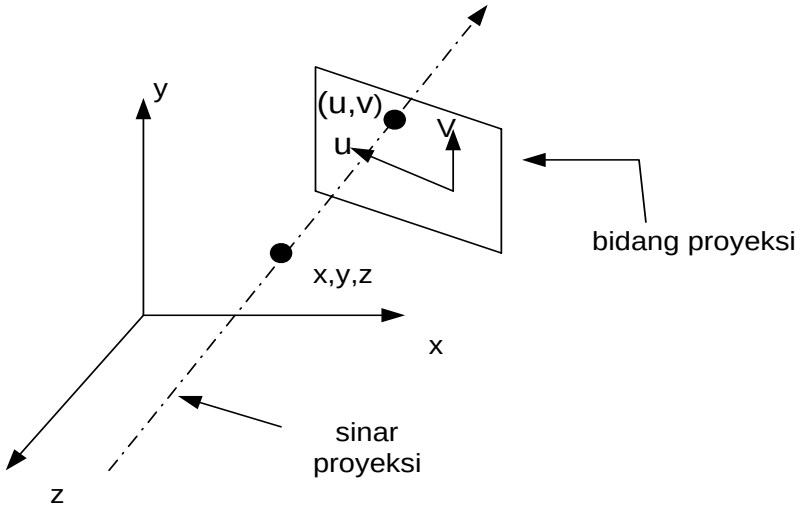
Mengingat ada 3 buah sumbu rotasi maka matriks transformasi yang digunakan juga bergantung kepada sumbu putar. Adapun isi matriks transformasi sesuai dengan sumbu putar didefinisikan sebagai 2.11

$$\begin{bmatrix} & \alpha & \alpha \\ & \alpha & \alpha \end{bmatrix} \quad \begin{bmatrix} \alpha & & \alpha \\ & & \\ \alpha & & \alpha \end{bmatrix} \quad \begin{bmatrix} \alpha & \alpha & \\ \alpha & \alpha & \\ & & \end{bmatrix} \tag{2.11}$$

2.6 Proyeksi

Proyeksi merupakan salah satu jenis transformasi , yaitu transformasi koordinat. Proyeksi merupakan proses dimana informasi tentang titik di sebuah sistem koordinat berdimensi n dipindahkan ke sistem koordinat berdimensi kurang dari n. sebagai contoh, titik (x,y,z) yang berada di sistem koordinat berdimensi 3 dipindahkan ke sistem koordinat yang berdimensi 2 sehingga menjadi (x,y), transformasi tersebut tentunya harus memperhitungkan pengaruh z terhadap titik (x,y). Proyeksi dapat dilakukan terhadap bidang datar (*planar*) atau kebidang kurva. Bab ini hanya akan membahas proyeksi ke bidang *planar* atau disebut sebagai *planar geometric projections.*

Planar geometric projections dilakukan melalui sinar proyeksi yang muncul dari titik pusat proyeksi melewati setiap titik dari benda dan memotong bidang proyeksi (*projection plane*) untuk mendapatkan benda hasil proyeksi (Gambar 2.7)



Gambar 2.7 Proyeksi Planar

Proyeksi *planar* dapat dibagi menjadi dua macam, yaitu: proyeksi ***parallel*** dan proyeksi ***perspektif***. Perbedaan utama antara kedua proyeksi tersebut adalah, pada proyeksi *perspektif* jarak antara titik pusat proyeksi ke bidang proyeksi bersifat *infinite* (tertentu) sedangkan pada proyeksi parallel jarak antara titik pusat proyeksi ke bidang proyeksi tidak terhingga.

2.6.1 Proyeksi paralel

Proyeksi parallel dapat dikategorikan menurut hubungan antara arah proyeksi dengan vektor normal dari bidang proyeksi, ke dalam dua macam proyeksi : *orthographic* dan *oblique.*

2.6.1.1 Proyeksi Orthographic

Proyeksi *Orthographic* diperoleh apabila sinar proyeksi tegak lurus dengan bidang proyeksi. Proyeksi *orthographic* sering digunakan untuk menghasilkan tampak depan, tampak belakang, tampak samping dan tampak atas dari sebuah benda atau disebut sebagai *Multiview orthographic*. Tampak atas, tampak belakang dan tampak dari samping sebuah benda sering disebut sebagai *elevation.* Sedangkan tampak dari atas disebut sebagai *plan view* (Hearn dan Baker, 2002).

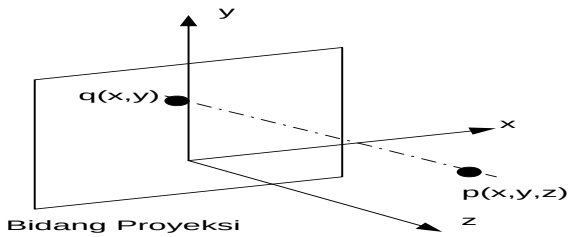
Transformasi untuk proyeksi *multiview orthographic* dapat diperoleh denga rumus 2.12

Proyeksi terhadap bidang x-z:qx = px,qy =pz

Proyeksi terhadap bidang y-z:qx = px,qy =pz

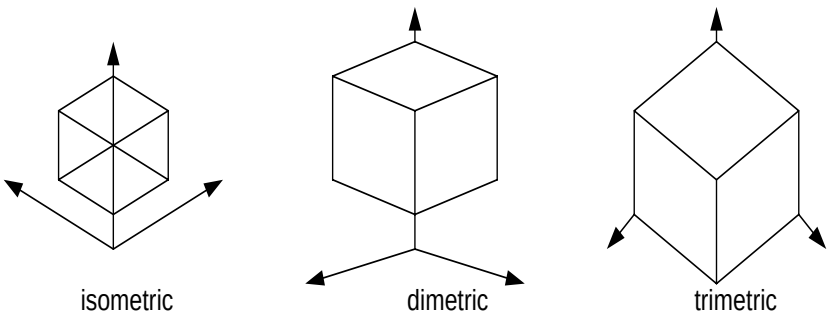
Proyeksi terhadap bidang x-y:qx = px,qy =py (2.12)

Dimana q(x,y) merupakan titik hasil proyeksi dari p(x,y,z) seperti digambarkan pada gambar 2.8



Gambar 2.8 transformasi untuk memperoleh proyeksi *orthographic*

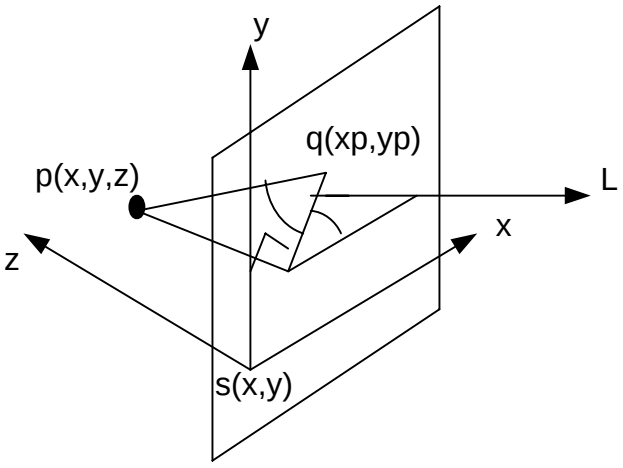
Proyeksi *orthographic* yang menampakan lebih dari satu permukaan benda disebut sebagai proyeksi *axonometric*. Apabila proyeksi *axonometric* dilakukan dengan mengatur agar bidang proyeksi berpotongan dengan ketiga sumbu koordinat (*principal axes*) pada sudut yang sama maka kita akan memperoleh proyeksi *isometric.* Jenis lain dari proyeksi a*xonometric* adalah proyeksi *dimetric* yaitu proyeksi yang diperoleh dengan mengatur agar bidang proyeksi berpotongan dengan dua sumbu utama pada sudut yang sama, sedangkan proyeksi *trimetric* diperoleh apabila ketiga sumbu utama berpotongan dengan bidang proyeksi pada sudut yang berbeda. Gambar II.9. memperlihatkan proyeksi *isometric*, *diametric* dan *trimetric*.



Gambar 2.9 proyeksi *axonometric*

2.6.1.2 Proyeksi Oblique

Proyeksi oblique diperoleh dengan cara membuat sinar proyeksi tidak tegak lurus terhadap bidang proyeksi. Proyeksi *oblique* membutuhkan dua buah sudut yaitu α dan β seperti seperti ditunjuakn pada Gambar 2.9. titik p(x,y,z) diproyeksikan menjadi titik q(xp,yp) di bidang proyeksi. Titik hasil proyeksi *orthographic* terletak di s(x,y) sinar proyeksi membuat sudut α terhadap garis q-s yang terletak di bidang proyeksi. Garis q-s dengan panjang L membentuk sudut-sudut terhadap arah mendatar dari bidang proyeksi.



Gambar 2.10 proyeksi *Oblique* dari titik p(x,y,z) ke titik q(xp,yp) (sumber : Hearn dan Baker, 2003)

Koordinat hasil proyeksi dapat dituliskan sebagai 2.13

$$Xp = X + L\cos\beta$$

$$Yp = Y + L\sin\beta \qquad (2.13)$$

Panjang L merupakan fungsi dari koordinat z dan dapat dihitung sebagai 2.14

$$\text{Tan}\,\alpha = \frac{z}{L} = \frac{1}{L1} \qquad (2.14)$$

Dengan L1 merupakan panjang dari q-s saat z = 1. rumus 2.11

$$\frac{z}{L} = \frac{1}{L1}$$

$$L = zL1 \qquad (2.15)$$

Sehingga rumus II.13 dapat ditulis ulang sebagai 2.16

$$Xp = x + z\,(L1\cos\beta)$$

$$Yp = y + z\,(L1\sin\beta) \qquad (2.16)$$

Apabila $\alpha = 90^0$ maka L1 = 0 sehingga dari rumus 2.16 kita memperoleh proyeksi orthographic, tetapi apabila L1 tidak sama dengan 0 maka kita akan memperoleh proyeksi oblique. Proyeksi oblique dengan $\alpha = 45^0$ disebut sebgai proyeksi cavalier, apabila $\alpha = 63{,}43495^0$ maka kita akan memperoleh proyeksi cabinet.

2.6.2 Proyeksi Perspektif

Proyeksi perspektif memberikan sudut pandang yang lebih realistis dibandingkan proyeksi orthographic. Proyeksi perspektif memberikan tampilan yang sama dengan apa yang kita lihat sehari-hari karena pada kenyataannya jarak benda terhadap kita akan mempengaruhi bagaimana benda tersebut terlihat. Benda yang terlihat jauh akan kelihatan kecil sedangkan benda yang dekat akan terlihat lebih besar. Efek ini disebut sebagai *shortening* (pemendekan).

Pada *perspektif* semua garis menghilang pada satu atau lebih titik yang sama atau disebut titik hilang (*vanishing point*). Hal ini mengakibatkan garis sejajar akan tampak tidak sejajar ketika diproyeksikan perspektif. Bergantung kepada

lokasi dimana kita melihat benda maka kita akan memperoleh efek : 1 titik hilang, 2 titik hilang dan 3 titik hilang.

2.7 Citra

Citra adalah fungsi dua dimensi yang terbentuk dari penglihatan dalam suatu tempat (*scane)* yang merupakan basis dari pembentukan citra.

Secara umum citra dibagi menjadi dua macam, yaitu :

1. Analog, yaitu citra hasil tangkapan lensa kamera terhadap tempat (*scane)* yang di-*scane* secara vertikal dan horizontal oleh *imager* kamera.
2. Digital, yaitu citra yang direpresentasikan sebagai sebuah matriks yang masing-masing elemennya merepresentasikan nilai intensitas.

2.7.1 Citra Digital

Citra digital dapat didefinisikan sebagai fungsi dua variabel, f(x,y), dimana x dan y adalah koordinat spasial dan nilai f(x,y) adalah intensitas citra pada koordinat tersebut. Teknologi dasar untuk menciptakan dan menampilkan warna pada citra digital berdasarkan pada penelitian bahwa sebuah warna merupakan kombinasi dari tiga warna dasar, yaitu merah, hijau, dan biru (*Red, Green, Blue - RGB*). Sebuah citra diubah ke bentuk digital agar dapat disimpan dalam memori komputer atau media lain. Proses mengubah citra ke bentuk digital bisa dilakukan dengan beberapa perangkat, misalnya scanner, kamera digital, dan handycam. Ketika sebuah citra sudah diubah ke dalam bentuk digital (selanjutnya disebut citra digital).

2.8 *Tresholding*

Thresohlding adalah proses perubahan *gray-level image* kedalam *bilevel image*. *Bilevel image* adalah suatu *image monochrome* yang terdiri dari *piksel* hitam dan putih. *Bilevel image* terdiri dari banyak informasi yang penting dari sebuah *image* (angka, posisi dan bentuk dari objek). Pengelompokan *image* oleh piksel *bilevel* dapat memperkecil dan memudahkan beberapa pengoperasian proses *image* seperti membentuk *recognition* dan *classification*.

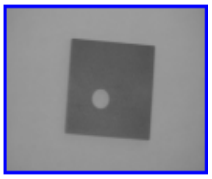
Secara keseluruhan nilai *bilevel*s menunjukan hasil yang dapat di klasifikasikan menjadi hitam(0) dan putih(1). Adanya kekasaran (*noise)* dan efek cahaya (*illumination)* akan mempengaruhi nilai yang dihasilkan dari pemrosesan.



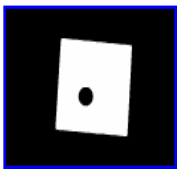
Gambar 2.11 Proses *Tresholding*

Masukkan dari operasi *tresholding* adalah gambar *grayscale* atau gambar berwarna. Pada implementasi sederhana, keluaran dari operasi *tresholding* adalah gambar binari yang diperlihatkan dengan pembagian (segmentasi). Piksel yang hitam dianggap sebagai *background* dan piksel putih dianggap sebagai *foreground.* Pada implementasi pembagian (segmentasi) sederhana ditentukan dengan satu parameter yang diketahui sebagai intensitas dari *threshold*. Pada satu nilai, piksel yang lain pada gambar dikompres dengan *threshold* ini. Jika

intensitas piksel lebih tinggi dari pada *threshold*, maka piksel akan diset sebagai warna putih pada keluarannya. Jika piksel lebih kecil dari *threshold*, maka akan diset sebagai warna hitam. Banyak *Threshold* dapat dispesifikasikan, jadi sekumpulan dari nilai intensitas dapat diset sebagai warna putih sementara yang lainnya diset sebagai warna hitam.



Gambar 2.12 Masukkan gambar *Thresholding*



Gambar 2.13 Hasil *Thresholding*

2.9 Kamera

Kamera merupakan salah satu hal yang penting dalam grafika tiga dimensi .Dengan kamera kita menentukan bagian *scene* dan berupa banyak objek dalam *scene* yang ditampilkan ke layer komputer. Kamera mempunyai 2 faktor penting yang harus diperhatikan yaitu:

a. Lokasi kamera

Lokasi kamera adalah sebuah titik dalam ruang yang didefinisikan dalam sistem koordinat XYZ. Lokasi kamera ini sangat berpengaruh pada proses rendering karena penggambaran harus dilakukan dari objek terjauh berurut sampai

objek terdekat. Hal ini disebabkan objek yang terjauh dari kamera dapat terhalang oleh objek yang lebih dekat.

b. Arah pandang Kamera

seperti halnya lokasi kamera arah pandang kamera yang didefinisikan dalam sistem koordinat XYZ. Yang dimaksud dengan arah pandang kamera adalah dari lokasi kamera yang sudah ditentukan. Kamera itu mengarah mana (yang menjadi fokus kamera dari lokasi itu apa)

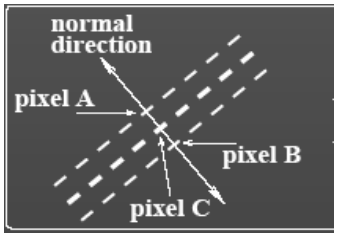
Kedua faktor diatas mempunyai peranan saling ketergantungan untuk proses rendering terhadap model-model yang akan ditampilkan pada layar komputer karena kamera dapat dianalogikan dengan mata manusia.

2.10 Pixel

Pixel atau piksel (*picture element /* unsur gambar*)* adalah tititk–titik kecil. Gambar apapun yang tampak di layar komputer sebenarnya tersusun dari titik-titik kecil yang disebut piksel.

Jika beberapa piksel diletakkan berderet, akan tampaklah suatu garis. Jadi semua garis, sehalus apapun tampaknya dilayar komputer, sebenarnya adalah deretan piksel.

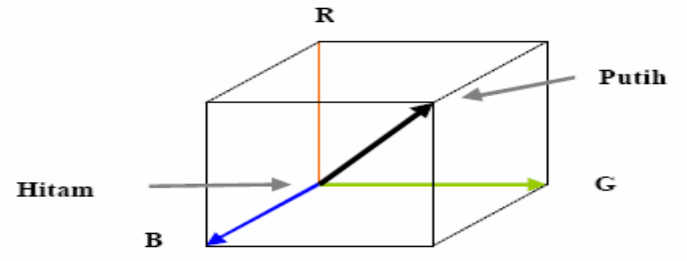
Sebuah piksel memang bisa dianggap sebagai sebuah titik, namun dalam kenyataannya, piksel-piksel lebih mirip dengan persegi panjang kecil yang tingginya tidak sebanding dengan lebarnya.



Gambar 2.14 piksel A, piksel B dan piksel C

2.11 Pengaturan Warna RGB

Red Green Blue (RGB) merupakan warna dasar yang ada pada sistem komputer. Data citra dapat dipisahkan menjadi komponen-komponen untuk masing-masing warna, yaitu merah *(red),* hijau *(green)* dan biru *(blue).* Warna tiap piksel ditentukan oleh kombinasi intensitas dari masing-masing komponen warna. Sebagai contoh, pada *RGB* 24 bit, masing-masing komponen warna dinyatakan dalam 8 *bit* atau 256 level. Contoh, warna biru langit direpsentasikan dengan R=181, G-189, dan B=249.



Gambar 2.15 Koordinat RGB

Misalnya citra dengan 8 bit per piksel mempunyai 256 warna dan citra dengan 24 bit mempunyai 32768 warna, jadi tiap piksel dinyatakan dengan ;

1. bit 0 sampai dengan 7 untuk warna merah.
2. bit 7 sampai dengan 15 untuk warna hijau
3. bit 16 sampai dengan 24 untuk warna biru

Kemungkinan kombinasi warna yang ada adalah = 2563 + 2562 + 2561 = *16.843.008*, dimana nilai 0 menyatakan warna hitam sedangakan nilai 16 843 008 menyatakan warna putih.