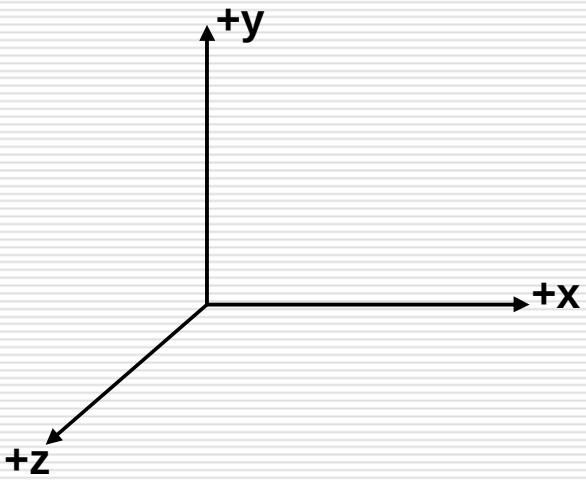
AIK21344
GRAFIKA DAN KOMPUTASI VISUAL

Chapter 6
Pengantar Grafika 3D

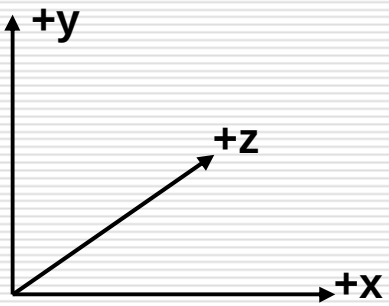
Pendahuluan

- Grafika Komputer dalam aplikasinya terbagi menjadi 2 :
  - Grafika 2D
  - Grafika 3D
- Aplikasi 2D banyak dipakai dalam pembuatan grafik, peta, kreasi 2D yang banyak membantu pemakai dalam membuat visualisasi.
- Grafika 2D memiliki kekurangan, yaitu : ketidakmampuannya untuk merepresentasikan objek 3D. Kekurangan ini sangat dirasakan terutama dalam bidang desain, dimana kebanyakan desainer membuat barang yang ada dalam dunia nyata yang berdimensi 3.
- Grafika 3D memiliki kemampuan untuk membuat visualisasi dari sebuah benda yang nyata yang dapat dilihat dari berbagai sudut pandang. Hal inilah yang membuat grafika 3D banyak dipakai terutama dalam bidang desain dari sebuah produk.

Sistem Koordinat 3D



Right-handed



Left-handed

Primitif 3D

- Dalam dunia 3D terdapat beberapa primitif seperti :
  - Titik (*point*)
  - Garis (*line*)
  - Bidang/Permukaan (*plane*/*surface*)
  - Bola (*sphere*)
  - Kubus(*cube*)
  - Silinder (*cylinder*)
  - Kerucut (*cone*)
  - Cincin (*torus*)
  - dll

Primitif 3D

- Titik
  - Posisi sebuah titik dalam grafika 3D diekspresikan dengan (x,y,z)
- Garis
  - Sebuah garis dibentuk dengan mendeskripsikan dua buah titik, yaitu (x1,y1,z1) dan (x2,y2,z2) yang sebagai ujung dari sebuah garis.
  - Sebuah garis dalam grafika 3D dapat diekspresikan dengan sepasang persamaan, yaitu :

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1}$$

$$\frac{z - z_1}{x - x_1} = \frac{z_2 - z_1}{x_2 - x_1}$$

Primitif 3D

- Bidang
  - Pada grafika 3D, terdapat sebuah geometri yang sangat penting, yaitu bidang datar (*plane*). Sebuah bidang datar pada grafika 3D dispesifikasikan dengan sebuah persamaan, yaitu :

$$Ax + By + Cz + D = 0$$

Representasi Object 3D

- Untuk merepresentasikan object 3D :
  - Persamaan Geometri
  - Constructive Solid Geometry (CSG)
  - Kurva & Permukaan Bezier
  - Lathe Object
  - Fractal

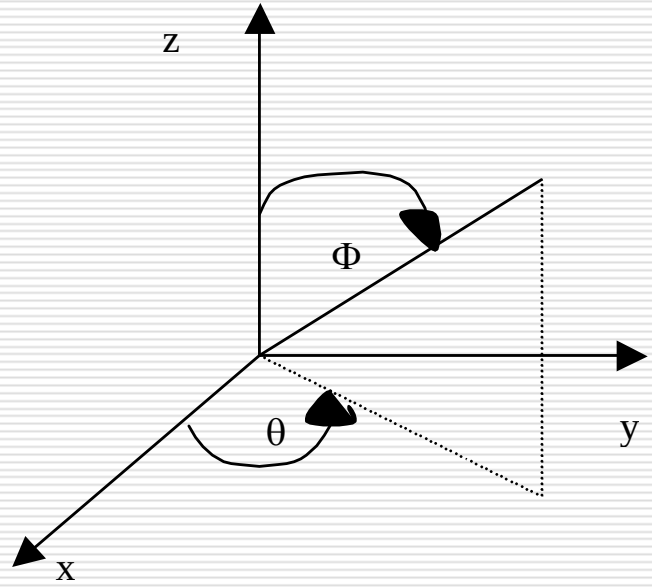
Representasi Object 3D

- Dengan Persamaan Geometri
  - Suatu object 3D dapat direpresentasikan langsung dengan menggunakan persamaan geometri dari object tersebut.
  - Misalkan : untuk membangun sebuah bola, maka bisa dengan menggunakan rumus :

$$X^2 + Y^2 + Z^2 = R^2$$

atau dengan rumus :

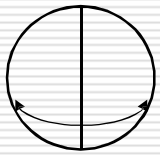
Representasi Object 3D



- $x = r.\sin\Phi.\sin\theta \quad ; 0 \leq \Phi \leq 2\pi$
- $y = r.\sin\Phi.\cos\theta \quad ; -\pi \leq \theta \leq \pi$
- $z = r.\cos\Phi$

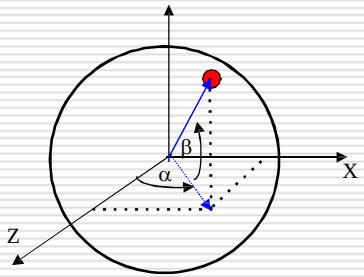
1. Menggambar Bola

Bola mempunyai koordinat khusus, dimana setiap titik pada bola mempunyai jarak yang sama terhadap titik pusatnya. Bola adalah hasil putar dari setengah lingkaran.



Gambar 1.1 : Bola adalah setengah lingkaran yang diputar

Pembentukan bola adalah:



Gambar 1.2: Pembentukan bola

Definisi koordinat titik-titik pada bola adalah:

$$x_{ij} = r.\cos(i\alpha).\sin(j\beta)$$

$$y_{ij} = r.\cos(i\alpha)$$

$$z_{ij} = r.\sin(i\alpha).\sin(j\beta)$$

Definisi face pada bola adalah:

$$p_0 = i.n + j$$
$$p_1 = (i+1).n + j$$
$$p_2 = (i+1).n + j + 1$$
$$p_3 = i.n + j + 1$$

Dan fungsi untuk membuat bola adalah:

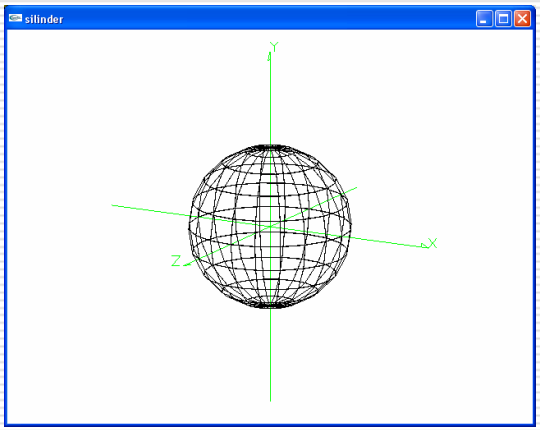
```
void createSphere(object3D_t &sphere,int n,float r){
float a=6.28/n;  float b=6.28/n;  int i,j;
sphere.NumberofVertices=(n+1)*n;  for(i=0;i<=n;i++){
for(j=0;j<n;j++){  sphere.pnt[i*n+j].x=r*cos(j*a)*sin(i*b);
sphere.pnt[i*n+j].y=r*cos(i*b);
sphere.pnt[i*n+j].z=r*sin(j*a)*sin(i*b);
}
}
sphere.NumberofFaces=n*n+2;  for(i=0;i<n;i++){
for(j=0;j<n;j++){
sphere.fc[i*n+j].NumberofVertices=4;
sphere.fc[i*n+j].pnt[0]=i*n+j;
sphere.fc[i*n+j].pnt[1]=(i+1)*n+j;
sphere.fc[i*n+j].pnt[2]=(i+1)*n+j+1;
sphere.fc[i*n+j].pnt[3]=i*n+j+1;  if(j==(n-1)){
sphere.fc[i*n+j].pnt[2]=i*n+j+1;  sphere.fc[i*n+j].pnt[3]=(i-
1)*n+j+1;
}
}
}
sphere.fc[n*n].NumberofVertices=n;  for(i=0;i<n;i++)
sphere.fc[n*n].pnt[i]=i;  sphere.fc[n*n+1].NumberofVertices=n;
for(i=0;i<n;i++)
sphere.fc[n*n+1].pnt[i]=(n+1)*n-1-i;  color_t c={1,1,0};
```

```
for(i=0;i<sphere.NumberofFaces;i++)
sphere.fc[i].col=c;   for(i=0;i<sphere.NumberofVertices;i++)
sphere.col[i]=c;
}
```

Dan perintah pada userdraw untuk menggambar bola adalah sebagai berikut:

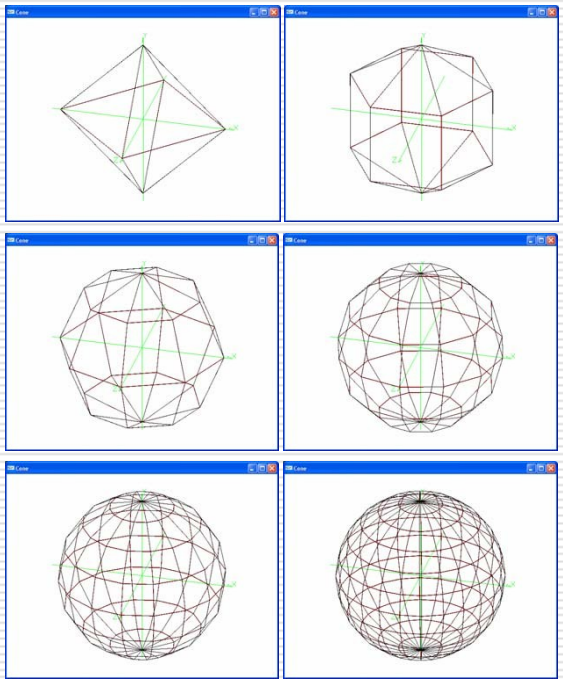
```
void userdraw(void){
matrix3D_t tilting=rotationXMTX(0.25)*rotationYMTX(-0.5);
setColor(0,1,0);
drawAxes(tilting);  object3D_t bola;  createSphere(bola,20,100);
setColor(0,0,0);  draw3D(bola,tilting);
}
```

Hasilnya adalah:



Gambar 1.3 Contoh bola dengan n=20

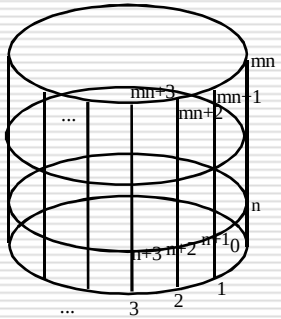
Gambar berikut menunjukkan pengaruh nilai jumlah titik pembentuk lingkaran n terhadap gambar bola yang dihasilkan oleh fungsi ini, yaitu dengan memberikan nilai n adalah 4, 6, 8, 12, 16, dan 24.



Gambar 1.4 Contoh bola dengan n=4, 6, 8, 12, 16 dan 24

2. Menggambar Silinder Bertumpuk

Silinder bertumpuk adalah obyek 3 dimensi yang berupa tumpukan silinder. Teknik pembuatannya hampir sama dengan teknik pembuatan silinder, tetapi obyek ini terdiri dari banyak silinder yang dijadikan satu.



Gambar 2.1. Silinder bertumpuk

Silinder bertumupuk dengan m silinder dan n titik pembenyuk lingkaran mempunyai r[] sebanyak m+1 dan h[] sebanyak m. Pembuatan silinder bertumpuk dengan m silinder dan n buah titik untuk menghasilkan lingkaran adalah:

(1)Pembuatan titik Jumlah titik = m×n

Koordinat titik-titik:

$x_i = r_i .\cos(i\theta)$

$y_i = b$ dengan $b = \sum_{j=1}^{i} h_j$

$z_i = r_i .\sin(i\theta)$

(2)Pembuatan Face Jumlah Face = m×n+2

m×n buah face untuk bagian tepi, 2 buah face untuk lingkaran atas dan bawah. Face bagian tepi:

Jumlah titik = 4

$p_0 = i.n + j$

$p_1 = (i+1).n + j$

$p_2 = (i+1).n + j + 1$

$p_3 = i.n + j + 1$

Face bagian alas:

Jumlah titik = n

$$p_i = i$$

Face bagian atas:

Jumlah titik = n

$$p_i = (m+1).n - i - 1$$

Sehingga fungsi untuk membuat silinder bertumpuk adalah sebagai berikut:

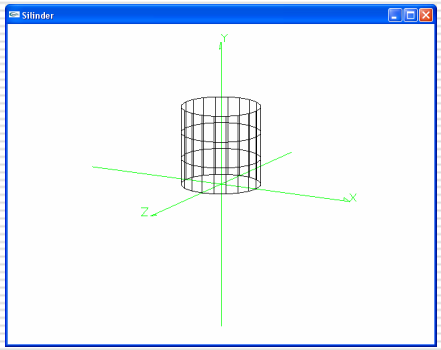
```
void createCylinderN (object3D_t &silinder,int m,int n,float r[],float h[]){
float a=6.26/n;  float b=0;
int i,j;
silinder.NumberofVertices=(m+1)*n;  for(i=0;i<=m;i++){
if(i>0) b=b+h[i-1];  for(j=0;j<n;j++){
silinder.pnt[i*n+j].x=r[i]*cos(j*a);
silinder.pnt[i*n+j].y=b;  silinder.pnt[i*n+j].z=r[i]*sin(j*a);
}
}
silinder.NumberofFaces=m*n+2;  for(i=0;i<m;i++){
for(j=0;j<n;j++){  silinder.fc[i*n+j].NumberofVertices=4;
silinder.fc[i*n+j].pnt[0]=i*n+j;  silinder.fc[i*n+j].pnt[1]=(i+1)*n+j;
silinder.fc[i*n+j].pnt[2]=(i+1)*n+j+1;
silinder.fc[i*n+j].pnt[3]=i*n+j+1;  if(j==(n-1)){
silinder.fc[i*n+j].pnt[2]=i*n+j+1;  silinder.fc[i*n+j].pnt[3]=(i-1)*n+j+1;
}
}
}
silinder.fc[m*n].NumberofVertices=n;  for(i=0;i<n;i++) silinder.fc[m*n].pnt[i]=i;
silinder.fc[m*n+1].NumberofVertices=n;  for(i=0;i<n;i++)
```

```
    silinder.fc[m*n+1].pnt[i]=(m+1)*n-1-i;
}
```

Contoh hasil dari silinder bertumpuk dengan m=3, n=20, r={60,60,60,60} dan h={40,40,40} adalah:

```
void userdraw(void){
matrix3D_t tilting=rotationXMTX(0.25)*rotationYMTX(-0.5);
setColor(0,1,0);
drawAxes(tilting);
float r[4]={60,60,60,60};
float h[3]={40,40,40};
object3D_t silinder;  createCylinderN(silinder,3,20,r,h);
setColor(0,0,0);  draw3D(silinder,tilting);
}
```

Hasilnya adalah sebagai berikut:



Gambar 2.2. Contoh silinder bertumpuk

Untuk menggambar silinder bertumpuk dituliskan:

createCylinderN(obyek3D, m, n, r, h)

Dimana:

Obyek3D menyatakan nama obyek 3D m

adalah jumlah silinder

n adalah jumlah titik pembentuk lingkaran alas

r adalah jari-jari pada setiap lingkaran, berupa array sejumlah m+1

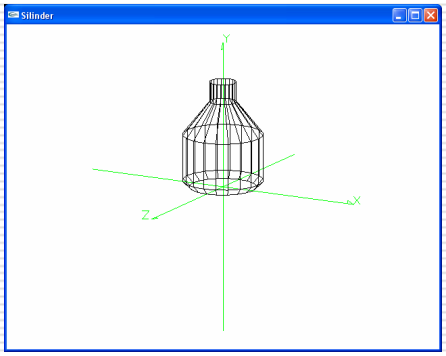
h adalah tinggi pada setiap lingkaran, berupa array sejumlah m

Sebagai contoh, silinder bertumpuk dengan 4 buah silinder, n=20,
r={50,60,60,20,20},

dan h={10,70,50,30} adalah:

```
void userdraw(void){
matrix3D_t tilting=rotationXMTX(0.25)*rotationYMTX(-0.5);
setColor(0,1,0);
drawAxes(tilting);
float r[5]={50,60,60,20,20};
float h[4]={10,70,50,30};
object3D_t silinder;  createCylinderN(silinder,4,20,r,h);
setColor(0,0,0);  draw3D(silinder,tilting);
}
```

Hasilnya adalah:



Gambar 2.3. Contoh hasil silinder bertumpuk

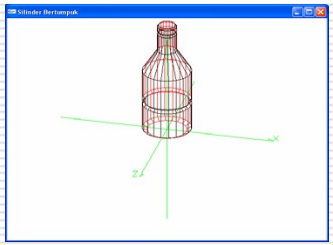
Model silinder bertumpuk ini sangat baik digunakan untuk menghasilkan obyek-obyek grafik 3 dimensi yang cukup menarik. Beberapa contoh hasil dari model silinder bertumpuk dengan mengatur nilai r dan h yang berupa array.

•Membuat botol

```
void userdraw(void){
```

```
matrix3D_t tilting=rotationXMTX(0.5)*rotationYMTX(-0.25);
setColor(0,1,0);  drawAxes(tilting);
object3D_t o;
float r[12]={40,50,50,45,50,50,20,20,17,20,20,17};
float h[11]={5,60,5,5,70,50,40,4,4,10,4};
createCylinderN(o,11,20,r,h);  draw3D(o,tilting);
}
```

Hasilnya adalah:

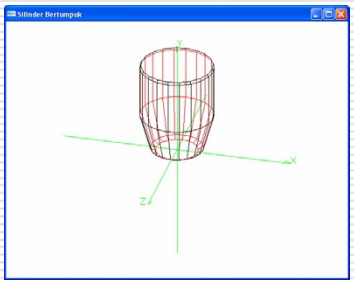


Gambar 2.4. Contoh gambar botol

- Membuat gelas

```
void userdraw(void){
matrix3D_t tilting=rotationXMTX(0.5)*rotationYMTX(-0.25);
setColor(0,1,0);
drawAxes(tilting);  object3D_t o;
float r[5]={40,50,70,70,68};
float h[4]={5,70,100,5};
createCylinderN(o,4,20,r,h);  draw3D(o,tilting);
}
```

Hasilnya adalah:

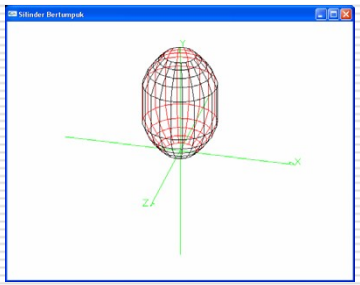


Gambar 2.5. Contoh gambar gelas

- Membuat lampion

```
void userdraw(void){
matrix3D_t tilting=rotationXMTX(0.5)*rotationYMTX(-0.25);
setColor(0,1,0);
drawAxes(tilting);  object3D_t o;
float r[10]={30,40,50,60,70,70,60,50,40,30};
float h[9]={10,12,14,25,80,25,14,12,10};
createCylinderN(o,9,20,r,h);  draw3D(o,tilting);
}
```

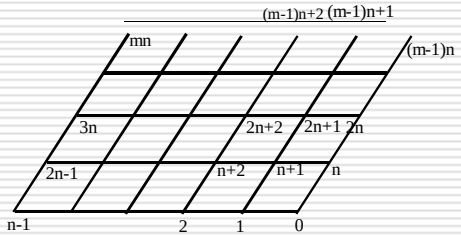
Hasilnya adalah:



Gambar 2.6 Contoh gambar lampion

7.8 Menggambar Papan Datar (Plane)

Papan datar (plane) adalah obyek 3 dimensi yang berupa bidang datar yang merupakan kumpulan dari kotak-kotak kecil seperti pada gambar 7.24 berikut.



Gambar 2.7 Papan bidang datar

Papan di atas terdiri dari m×n buah persegi empat kecil dengan ukuran dx dan dy dan dituliskan dengan:

createPlane(obyek3D, m, n, dx, dy);

Untuk membuat papan, prosesnya adalah:

(1)Penentuan koordinat titik: Titik ke (i,j) adalah:

x = j . dx y = 0

z = i . dy

(2)Penentuan face

Untuk face ke (i,j) adalah:

i.n+j (i+1).n+j (i+1).n+j+1
i*n+j+1

Sehingga fungsi untuk membuat papan datar di atas adalah sebagai berikut:

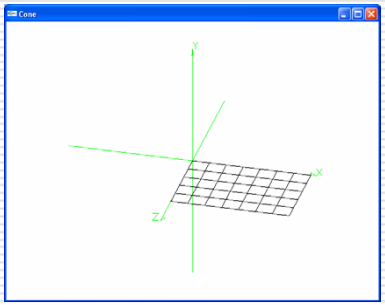
```
void createPlane(object3D_t &papan,int m, int n, float dx, float dy) {
int i,j,k;  for(i=0;i<m;i++)
for(j=0;j<n;j++){
k=i*n+j;  papan.pnt[k].x=j*dx;  papan.pnt[k].y=0;  papan.pnt[k].z=i*dy;
}
```

```
papan.NumberofVertices=m*n;
for(i=0;i<m-1;i++)
for(j=0;j<n-1;j++){
k=i*(n-1)+j;  papan.fc[k].NumberofVertices=4;
papan.fc[k].pnt[0]=i*n+j;  papan.fc[k].pnt[1]=(i+1)*n+j;
papan.fc[k].pnt[2]=(i+1)*n+j+1;
papan.fc[k].pnt[3]=i*n+j+1;
}
papan.NumberofFaces =(m-1)*(n-1);
}
```

Dan untuk membuat papan dengan m=6, n=8, dx=30 dan dy=30, pada userdraw dituliskan sebagai berikut:

```
void userdraw(void){
matrix3D_t tilting=rotationXMTX(0.5)*rotationYMTX(-0.25);
setColor(0,1,0);
drawAxes(tilting);  object3D_t o;  createPlane(o,6,8,30,30);
setColor(0,0,0);
draw3D(o,tilting);
}
```

Hasilnya adalah:



Gambar 2.7. Contoh papan datar

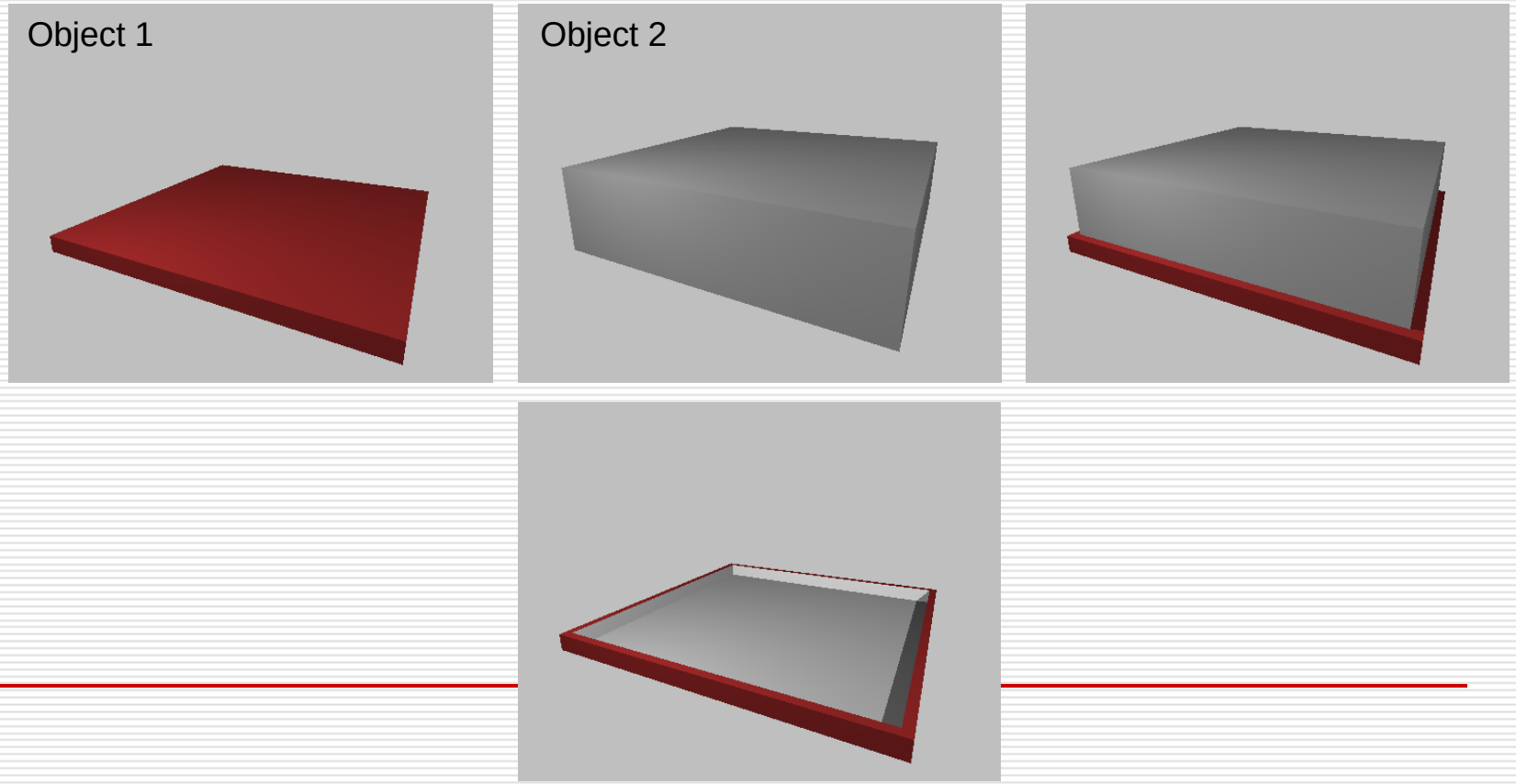
Obyek-obyek 3 dimensi yang sudah dibahas di atas adalah obyek-obyek dasar geometri. Untuk obyek-obyek yang lain bisa dibangun dengan menggabungkan obyek- obyek di atas atau melakukan transformasi bentuk seperti *lofting* (mengubah model kurva).

Representasi Object 3D

- Dengan Constructive Solid Geometry (CSG)
  - CSG adalah suatu cara membentuk object dengan jalan menggabungkan atau memotong(mengurangi) dari beberapa object primitif 3D.
  - CSG dalam POV-Ray melibatkan :
    - difference
    - union
    - intersect

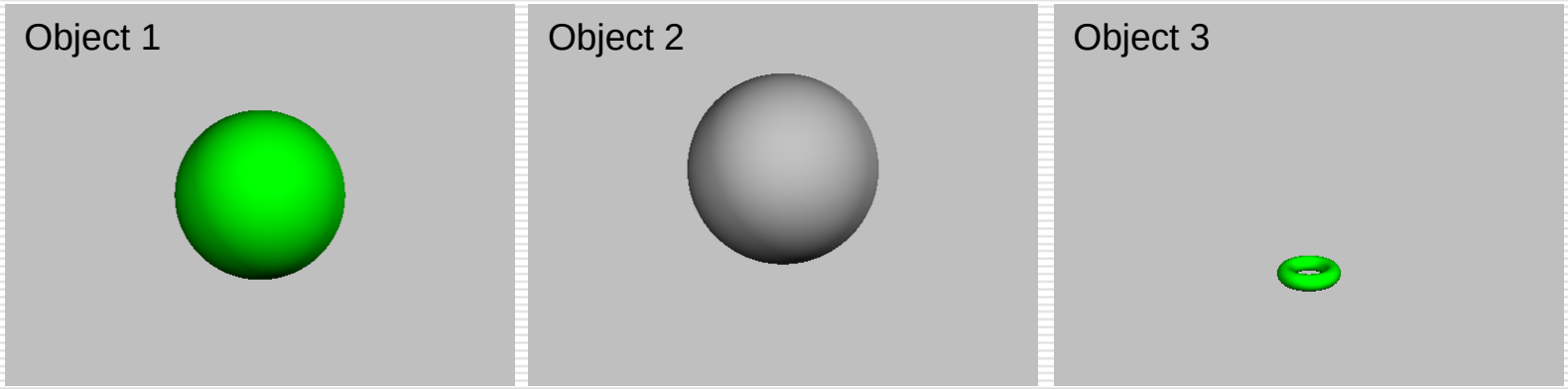
Representasi Object 3D

- Dengan CSG – cont.
  - Contoh 1:



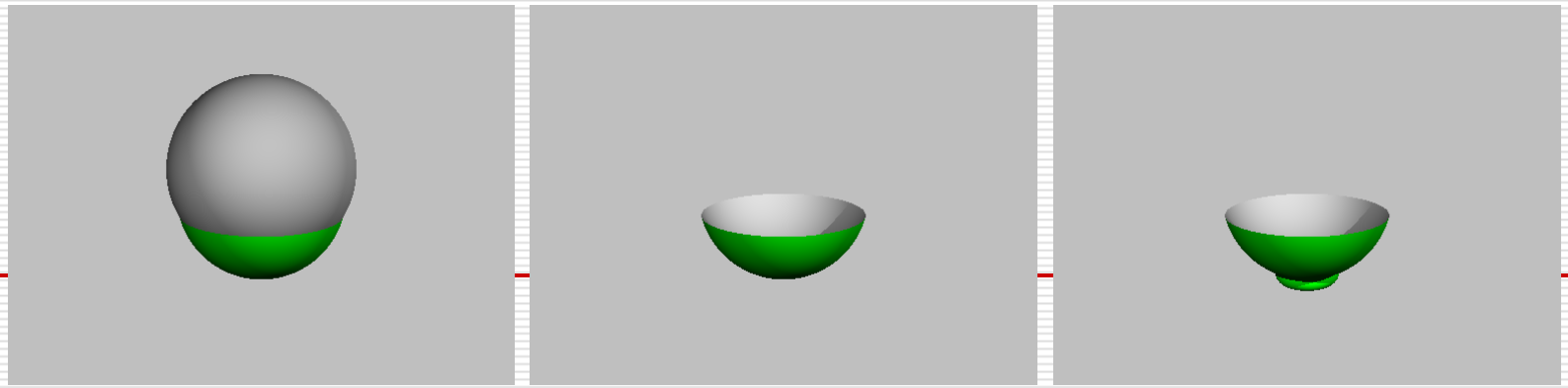
Representasi Object 3D

- Dengan CSG – cont.
  - Contoh 2:



Object 5 = Object 1 – Object 2

Object 6 = Object 5 + Object 3



Representasi Object 3D

- Kurva & Permukaan Bezier

$$x(t) = \sum_{i=0}^{N} x_i . B_{i,N}(t)$$

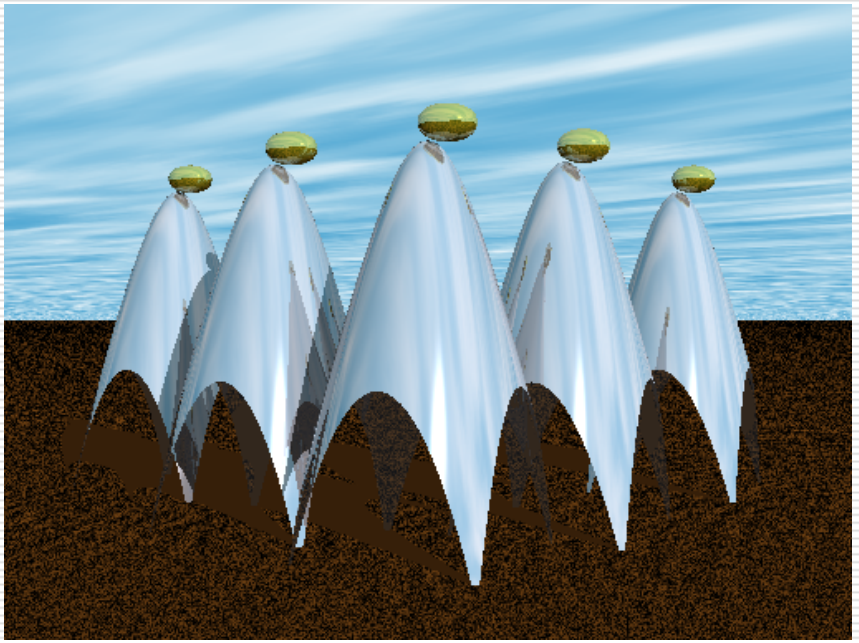
$$y(t) = \sum_{i=0}^{N} y_i . B_{i,N}(t)$$

$$z(t) = \sum_{i=0}^{N} z_i . B_{i,N}(t)$$

dimana :

$$B_{i,N} = C(N,i).(1-t)^{N-i} t^i$$

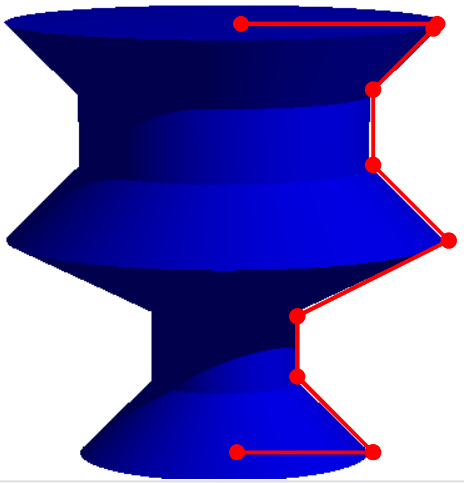
$$C(N,i) = \frac{N!}{i!.(N-i)!}$$



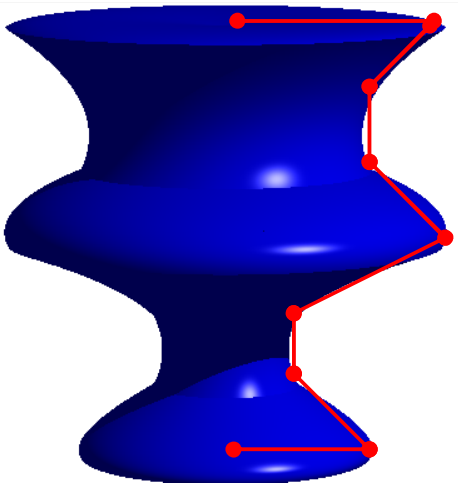
Representasi Object 3D

- Lathe Object

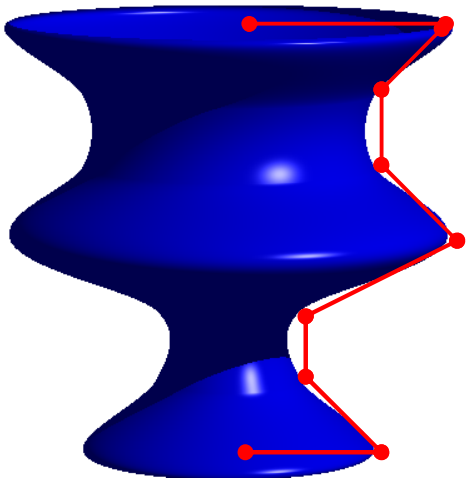
Linear Spline



Quadratic Spline



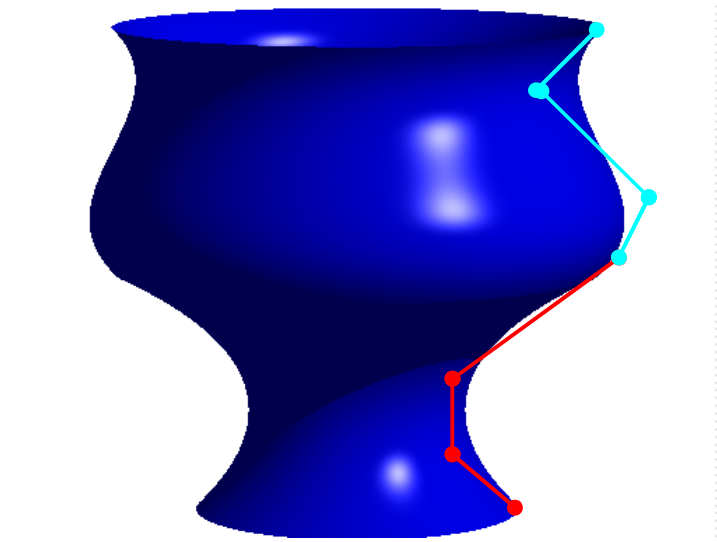
Cubic Spline



Representasi Object 3D

Lathe Object - cont

Bezier Spline



Representasi Object 3D

- Dengan Fractal