Menggambar Plot 3D dengan EMT

Ini adalah pengenalan untuk plot 3D pada Euler. Kita membutuhkan plot 3D untuk memvisualisasikan sebuah fungsi dari dua variabel.

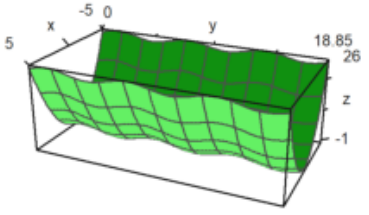
Euler menggunakan fungsi ini untuk algoritma pengurutan untuk menyembunyikan bagian dalam latar belakang. Umumnya, Euler menggunakan proyeksi sentral. Secara bawaan dari kuadran x-y positif melewati titik asal x=y=z=0, tetapi sudut=0° terlihat arah dari sumbu-y. Pandangan sudut dan tinggi dapat berubah.

Euler dapat memplotkan

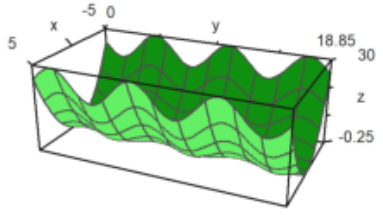
- permukaan dengan bayangan dan garis level atau rentang level
- titik titik yang tersebar
- kurva parametrik
- permukaan implisit

Sebuah plot 3D dari sebuah fungsi menggunakan plot3d. Cara paling mudah untuk membuatnya adalah dengan ekspresi dalam x dan y. Parameter r dapat diatur dalam rentang dari sekeliling plot (0, 0).

```
>aspect(1.5); plot3d("x^2+sin(y)",-5,5,0,6*pi):
```




```
>plot3d("x^2+x*sin(y)",-5,5,0,6*pi):
```



Silakan lakukan modifikasi agar gambar "talang bergelombang" tersebut tidak lurus melainkan melengkung/melingkar, baik melingkar secara mendatar maupun melingkar turun/naik (seperti papan peluncur pada kolam renang. Temukan rumusnya.

Fungsi fungsi dari Dua Variabel

Untuk membuat grafik dari sebuah fungsi, gunakan

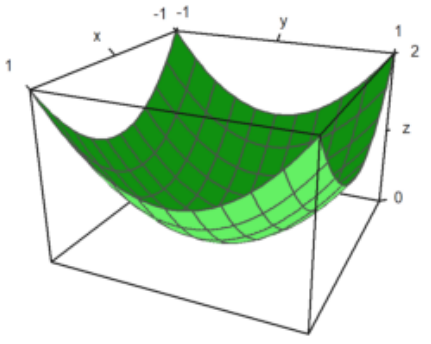
- sebuah ekspresi sederhana dalam x dan y,
- nama fungsi dari dua variabel
- atau matriks data

Secara bawaan adalah kabel yang terisi dengan warna-warna yang berbeda dalam kedua sisi. Catatan bahwa nomor bawaan dari interval grid adalah 10, tetapi plot menggunakan angka bawaan dari persegi 40x40 untuk membuat permukaan. Ini dapat diubah.

- n=40, n=[40,40]: banyaknya dari garis grid dalam setiap arah
- grid=10, grid=[10,10]: banyaknya garis grid dalam setiap arah

Kami menggunakan bawaan n = 40 dan grid = 40.

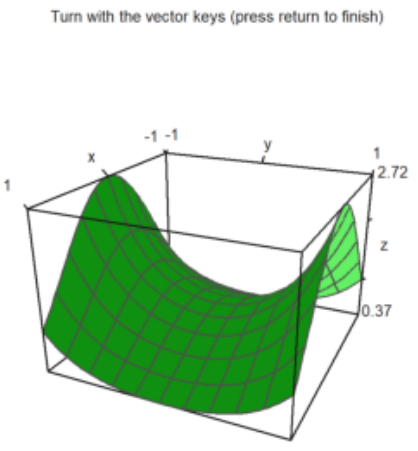
```
>plot3d("x^2+y^2"):
```



Interaksi pengguna dapat memungkinkan dengan mengatur parameter >user. Pengguna dapat menekan tombol berikut.

- left,right,up,down: mengubah pandangan sudut
- +,-: memperbesar atau mengecilkan
- a: mengeluarkan sebuah anaglyph (lihat bawah)
- l: mengalihkan asal cahaya (lihat bawah)
- space: atur ulang ke bawaan
- return: selesaikan interaksi

```
>plot3d("exp(-x^2+y^2)",>user, ...
>  title="Turn with the vector keys (press return to finish)"):
```



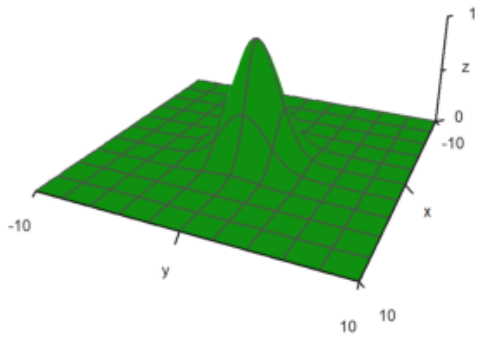
Plot selang untuk fungsi dapat dispesifikan dengan

- a,b: selang-x
- c,d: selang-y
- r: sebuah persegi sekitar yang simetri (0,0)
- n: banyaknya subinterval untuk plot

Ini beberapa parameter untuk menskalakan fungsi atau mengubah tampak dari grafik.

fscale: menskalakan fungsi nilai (bawaan <fscale)
scale: angka atau vektor 1x2 untuk menskalakan ke arah x atau y
frame: tipe frame (bawaan 1)

```
>plot3d("exp(-(x^2+y^2)/5)",r=10,n=80,fscale=4,scale=1.2,frame=3,>user):
```



Pandangan dapat diubah dengan banyak cara.

- distance: arah pandangan ke plot.
- zoom: nilai pembesaran.
- angle: arah ke sumbu-y negatif dalam radian.
- height: the height of the view in radians.

Nilai bawaan dapat diinspeksi atau diubah dengan fungsi view(). Ini akan mengembalikan parameter dalam urutan diatas.

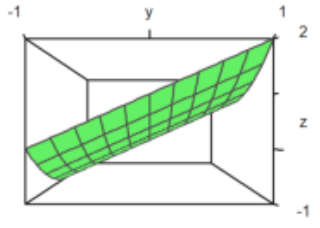
```
>view
```

```
[5,  2.6,  2,  0.4]
```

Jarak terdekat membutuhkan pembesaran yang lebih sedikit. Efeknya kira kira seperti sebuah lensa sudut yang lebar.

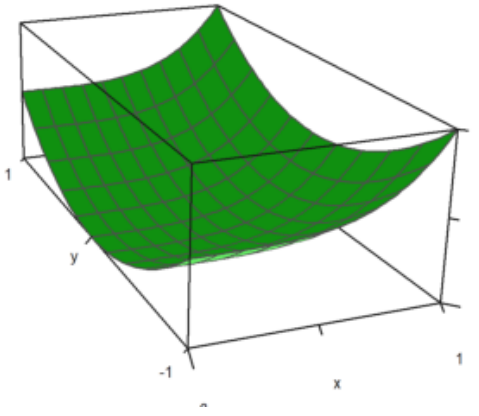
Seperti contoh berikut, angle=0 dan height=0 terlihat dari sumbu-y negatif. Label sumbu untuk y disembunyikan dalam kasus ini.

```
>plot3d("x^2+y",distance=3,zoom=1,angle=pi/2,height=0):
```



Plot selalu terlihat ke titik tengah dari kubus plot. Kamu dapat menggerakkan tengah dengan parameter center.

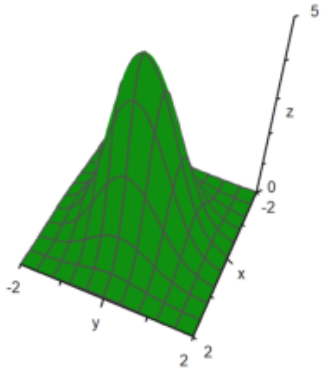
```
>plot3d("x^4+y^2",a=0,b=1,c=-1,d=1,angle=-20°,height=20°, ...
>  center=[0.4,0,0],zoom=5):
```



Plot dapat diskalakan untuk menyesuaikan ke bagian kubus untuk dilihat. Jadi ini tidak membutuhkan perubahan jarak atau memperbesar tergantung ukuran dari plot. Label ini merujuk pada ukuran sebenarnya.

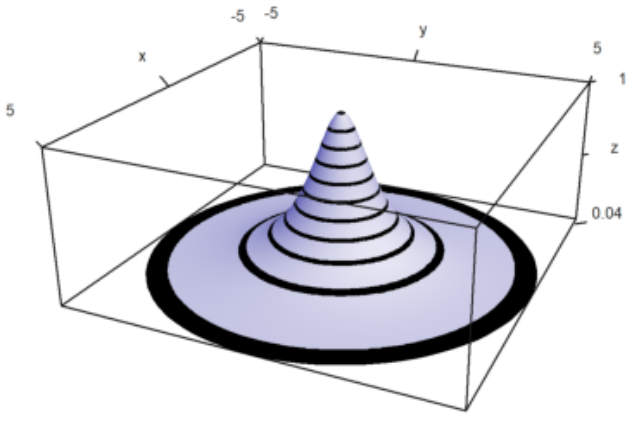
Jika kamu mematikan dengan scale=false, kamu harus berhati-hati, yang mana plot masih menyesuaikan ke jendela plot, dengan mengubah pandangan arah atau perbesaran dan menggerakkan titik tengah.

```
>plot3d("5*exp(-x^2-y^2)",r=2,<fscale,<scale,distance=13,height=50°, ...
>  center=[0,0,-2],frame=3):
```

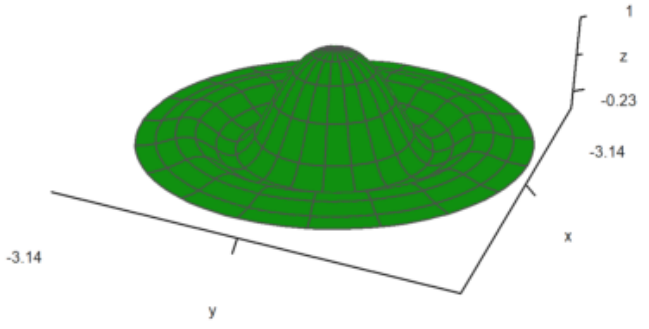


Plot kutub juga tersedia. Parameter polar=true menggambar sebuah plot polar. Fungsi harus tetap sebagai fungsi dari x dan y. Parameter "fscale" menskalakan fungsi dengan skala tersendiri. Sebaliknya fungsi diskalakan mengikuti sebuah kubus.

```
>plot3d("1/(x^2+y^2+1)",r=5,>polar, ...
>fscale=2,>hue,n=100,zoom=4,>contour,color=blue):
```



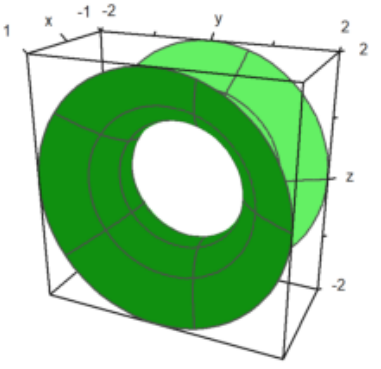

```
>function f(r) := exp(-r/2)*cos(r); ...
>plot3d("f(x^2+y^2)",>polar,scale=[1,1,0.4],r=pi,frame=3,zoom=4):
```



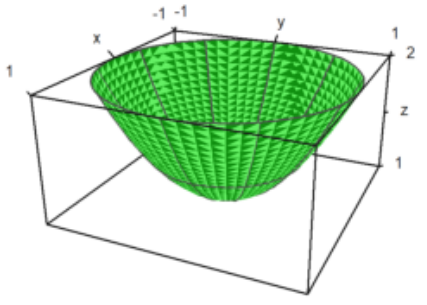
Parameter rotate memutar sebuah fungsi dalam x mengelilingi sumbu-x.

- rotate=1: Menggunakan sumbu-x
- rotate=2: Menggunakan sumbu-z

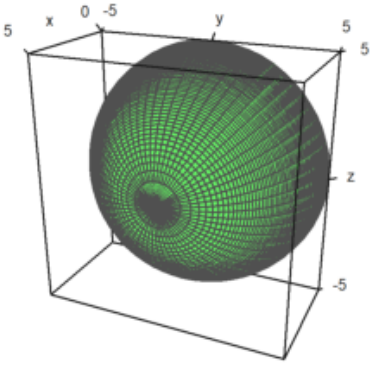
```
>plot3d("x^2+1",a=-1,b=1,rotate=true,grid=5):
```



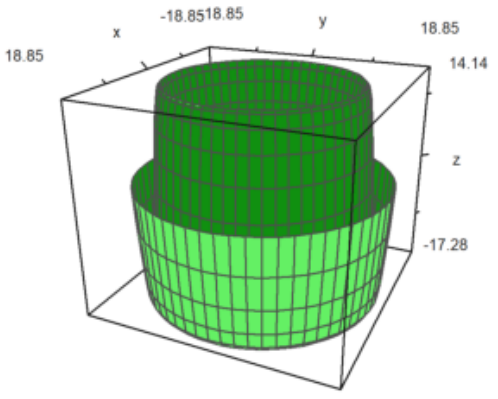

```
>plot3d("x^2+1",a=-1,b=1,rotate=2,grid=5):
```




```
>plot3d("sqrt(25-x^2)",a=0,b=5,rotate=1):
```

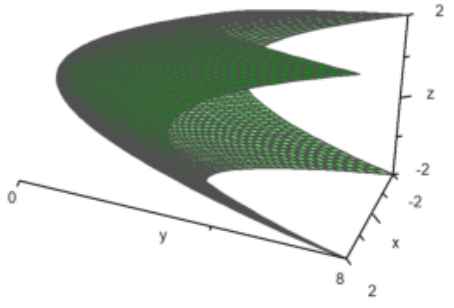



```
>plot3d("x*sin(x)",a=0,b=6pi,rotate=2):
```



Ini adalah plot dengan tiga fungsi.

```
>plot3d("x","x^2+y^2","y",r=2,zoom=3.5,frame=3):
```


2
z
-2
-2
x
2
8
y
0

Plot Kontur

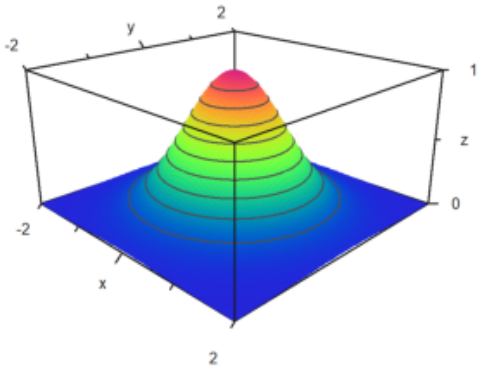
Untuk plot, Euler menambahkan garis bata. Ini memungkinkan untuk menggunakan level garis dan sebuah-warna hue atau sebuah spektral berwarna hue. Euler dapat menggambar tinggi dari fungsi dalam sebuah plot dengan bayangan. Dalam semua plot 3D Euler dapat menghasilkan anaglyph merah/cyan.

- >hue: Menggubah bayangan cahaya daripada garis.
- >contour: Membuat plot garis kontur otomatis dalam sebuah plot.
- level=... (or levels): Sebuah vektor dari nilai untuk garis kontur.

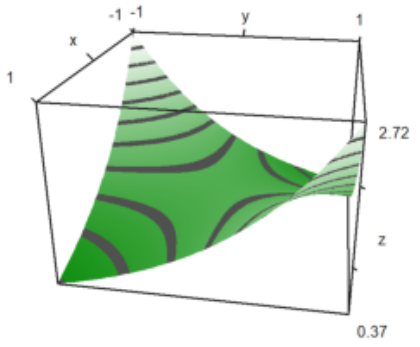
Bawaannya level=”auto”, yang mana menghitung beberapa garis level secara otomatis. Seperti yang kamu lihat dalam plot, level dalam rentang level.

Gaya bawan dapat diubah. Untuk plot kontur, kami menggunakan sebuah batasan yang halus untuk titik titik 100x100, menskalakan fungsi dan plot, dan menggunakan sudut yang berbeda dari pandangan.

```
>plot3d("exp(-x^2-y^2)",r=2,n=100,level="thin", ...
> >contour,>spectral,fscale=1,scale=1.1,angle=45°,height=20°):
```




```
>plot3d("exp(x*y)",angle=100°,>contour,color=green):
```

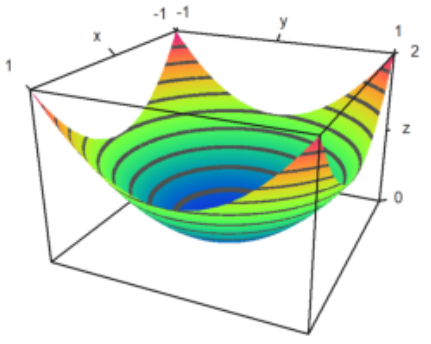


Bayangan bawaan menggunakan warna abu-abu. Tetapi selang spektral dari warna juga tersedia.

- >spectral: Digunakan skema spektral bawaan
- color=...: Menggunakan warna spesial atau skema spektral

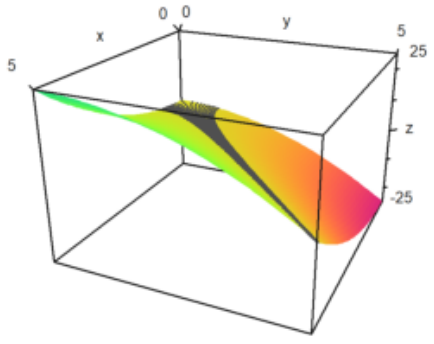
Untuk plot berikut, kami menggunakan skema spektral bawaan dan meningkatkan angka dari titik untuk mendapatkan tampilan yang halus.

```
>plot3d("x^2+y^2",>spectral,>contour,n=100):
```



Selain garis level otomatis, kami dapat juga mengatur nilai dari garis level. Ini akan menghasilkan garis level yang tipis daripada rentang level.

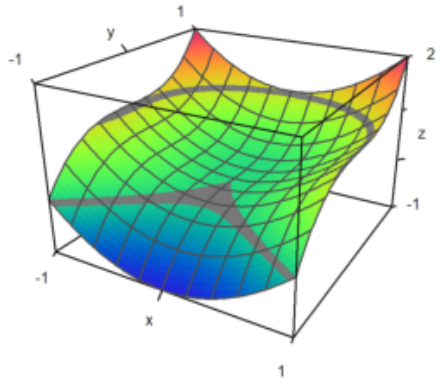
```
>plot3d("x^2-y^2",0,5,0,5,level=-1:0.1:1,color=redgreen):
```



Plot berikut, kami menggunakan dua level yang sangat luas dari -0.1 ke 1 dan dari 0.9 ke 1. Ini dimasukkan sebagai sebuah matrik dengan level batas seperti kolom.

Selebihnya, kami melapisi sebuah batasan dengan 10 interval pada setiap arah.

```
>plot3d("x^2+y^3",level=[-0.1,0.9;0,1], ...
>  >spectral,angle=30°,grid=10,contourcolor=gray):
```

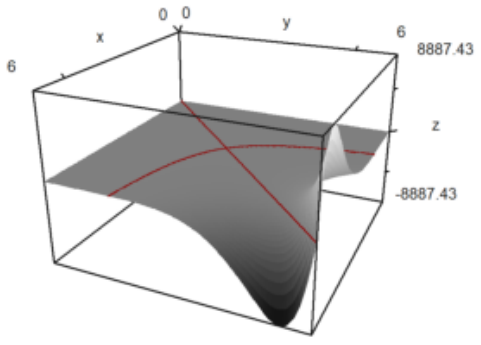


Contoh berikut, kami mengatur plot, dimana

$$f(x,y) = x^y - y^x = 0$$

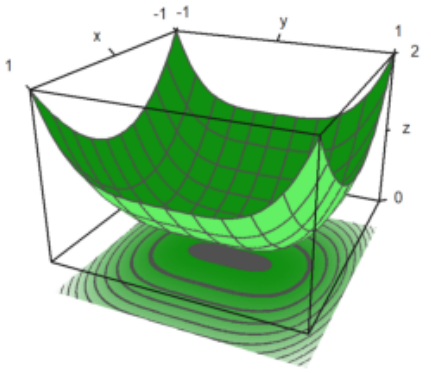
Kami menggunakan garis tipis tunggal untuk garis level.

```
>plot3d("x^y-y^x",level=0,a=0,b=6,c=0,d=6,contourcolor=red,n=100):
```



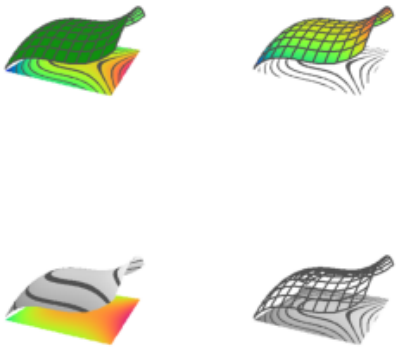
Ini memungkinkan untuk menampilkan sebuah bidang kontur dibawah plot. Warna dan jarak ke plot dapat di spesifikan.

```
>plot3d("x^2+y^4",>cp,cpcolor=green,cpdelta=0.2):
```



Ini beberapa gaya yang lainnya. Kami selalu mematikan frame dan menggunakan skema warna yang bervariasi untuk plot dan garis batas.

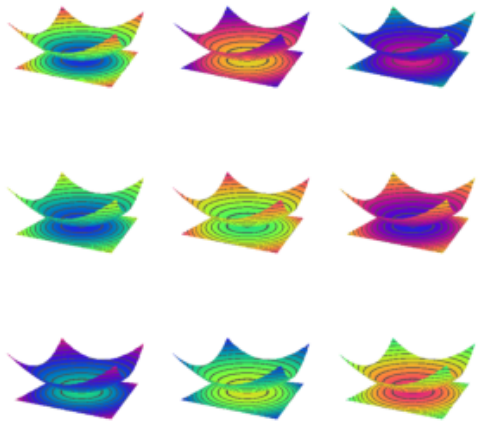
```
>figure(2,2); ...
>expr="y^3-x^2"; ...
>figure(1);  ...
>  plot3d(expr,<frame,>cp,cpcolor=spectral); ...
>figure(2);  ...
>  plot3d(expr,<frame,>spectral,grid=10,cp=2); ...
>figure(3);  ...
>  plot3d(expr,<frame,>contour,color=gray,nc=5,cp=3,cpcolor=greenred); ...
>figure(4);  ...
>  plot3d(expr,<frame,>hue,grid=10,>transparent,>cp,cpcolor=gray); ...
>figure(0):
```



Ini beberapa skema spektral yang lainnya, dihitung dari 1 ke 9. Tetapi kamu juga menggunakan color=value, dimana value

- spectral: untuk sebuah selang dari biru ke merah
- white: untuk selang pewarna
- yellowblue,purplegreen,blueyellow,greenred
- blueyellow, greenpurple,yellowblue,redgreen

```
>figure(3,3); ...
>for i=1:9;  ...
>  figure(i); plot3d("x^2+y^2",spectral=i,>contour,>cp,<frame,zoom=4);  ...
>end; ...
>figure(0):
```

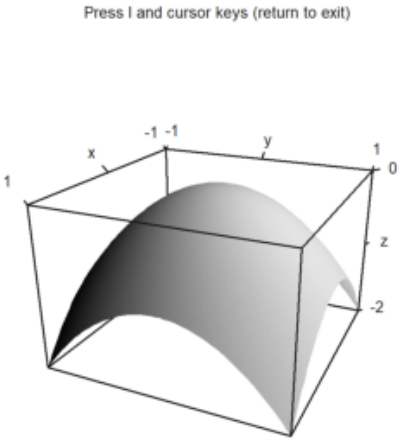


Sumber cahaya dapat diubah dengan 1 dan kunci kursor selama interaksi pengguna. Ini juga dapat diubah dengan parameter.

- light: arah cahaya
- amb: cahaya sekitar antara 0 dan 1

Catatan bahwa program tidak membuat perbedaan antara sisi dari plot. Tidak ada bayangan. Untuk hal ini akan membutuhkan Povray.

```
>plot3d("-x^2-y^2", ...
>  hue=true,light=[0,1,1],amb=0,user=true, ...
>  title="Press l and cursor keys (return to exit)"):
```

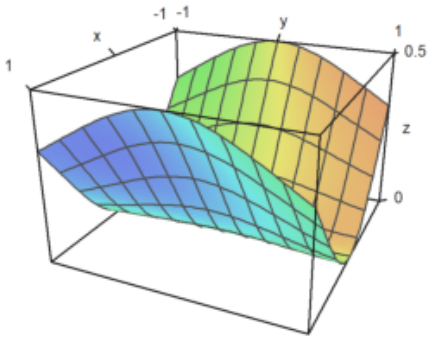


Parameter color mengubah warba dari permukaan. Warna dari garis level juga dapat diubah.

```
>plot3d("-x^2-y^2",color=rgb(0.2,0.2,0),hue=true,frame=false, ...
>  zoom=3,contourcolor=red,level=-2:0.1:1,dl=0.01):
```

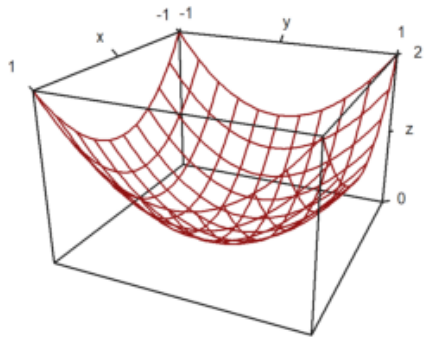
Color 0 memberikan spesial pelangi efek.

```
>plot3d("x^2/(x^2+y^2+1)",color=0,hue=true,grid=10):
```



Permukaan juga dapat transparan.

```
>plot3d("x^2+y^2",>transparent,grid=10,wirecolor=red):
```


-1
-1
x
y
1
1
2
z
0

Plot Implisit

Ini juga plot implisit dalam dimensi tiga. Euler menghasilkan potongan melalui objek. Fitur dari plot3d mencakup plot implisit. Plot ini menunjukkan penggaturan 0 dari fungsi dalam variabel 3.

Solusi dari

$$f(x,y,z)=0$$

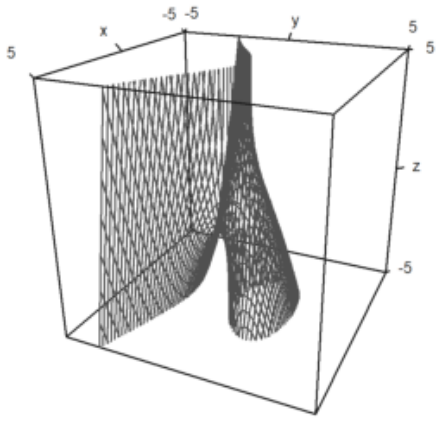
dapat divisualisasikan dalam potongan paralel ke bidang x-y-, x-z- dan y-z.

- implicit=1: memotong secara paralel ke bidang y-z
- implicit=2: memotong secara paralel ke bidang x-z
- implicit=4: memotong secara paralel ke bidang x-y

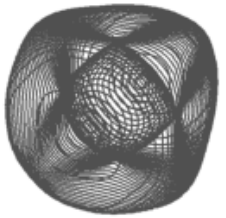
Menambahkan nilai-nilai ini, jika kamu suka. Dalam contoh ini kami membuat plot

$$M=\{(x,y,z):x^2+y^3+zy=1\}$$

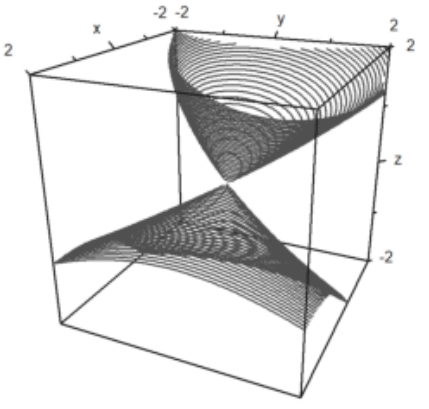
```
>plot3d("x^2+y^3+z*y-1",r=5,implicit=3):
```




```
>c=1; d=1;
>plot3d("((x^2+y^2-c^2)^2+(z^2-1)^2)*((y^2+z^2-c^2)^2+(x^2-1)^2)*((z^2+x^2-c^2)^2+(y^2-1)^2)-d",r=2,
```



```
>plot3d("x^2+y^2+4*x*z+z^3",>implicit,r=2,zoom=2.5):
```


x
-2
-2
y
2
2
2
z
-2

Membuat plot data 3D

Seperti plot2d, plot3d menerima data. Untuk objek 3D, kamu harus menyediakan sebuah matriks dari nilai x, y dan z, atau fungsi tiga atau ekspresi fx(x, y), fy(x, y), fz(x, y).

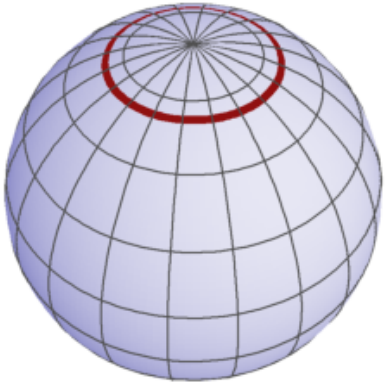
$$\gamma(t,s) = (x(t,s), y(t,s), z(t,s))$$

Karena x, y, z merupakan matriks, kami mengasumsikan bahwa (t, s) melewati sebuah batasan persegi. Sebagai hasilnya, kamu dapat membuat gambar persegi dalam ruang.

Kamu dapat menggunakan bahasa matriks Euler untuk menghasilkan koordinat secara efektif.

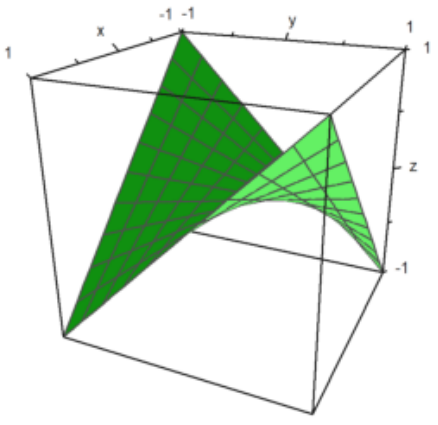
Seperti contoh berikut, kami menggunakan vektor dari nilai t dan sebuah kolom vektor dari nilai s untuk memparameterisasi wilayah dari bola. dalam penggambaran kami menandai wilayah-wilayah, dalam kasus kami wilayah polar.

```
>t=linspace(0,2pi,180); s=linspace(-pi/2,pi/2,90)'; ...
>x=cos(s)*cos(t); y=cos(s)*sin(t); z=sin(s); ...
>plot3d(x,y,z,>hue, ...
>color=blue,<frame,grid=[10,20], ...
>values=s,contourcolor=red,level=[90°-24°;90°-22°], ...
>scale=1.4,height=50°):
```



Ini sebuah contoh, yang mana grafik dari sebuah fungsi.

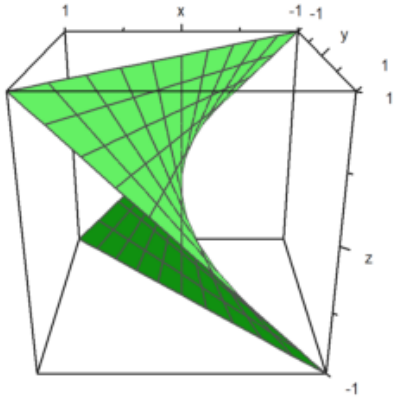
```
>t=-1:0.1:1; s=(-1:0.1:1)'; plot3d(t,s,t*s,grid=10):
```



Namun, kamu juga membuat semua pengurutan dari wilayah. Ini wilayah yang sama seperti fungsi

$$x = y\,z$$

```
>plot3d(t*s,t,s,angle=180°,grid=10):
```



Dengan usaha yang lebih, kami menghasilkan lebih banyak permukaan.

Seperti contoh berikut kami membuat pendangan bayangan untuk bola terdistorsi. Koordinat biasa untuk bola adalah

$$\gamma(t,s) = (\cos(t)\cos(s), \sin(t)\sin(s), \cos(s))$$

dengan

$$0 \leq t \leq 2\pi, \quad \frac{-\pi}{2} \leq s \leq \frac{\pi}{2}.$$

Kami mendistorsi ini dengan sebuah faktor

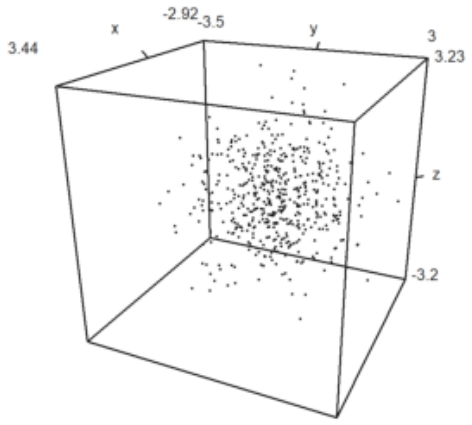
$$d(t,s) = \frac{\cos(4t) + \cos(8s)}{4}.$$

```
>t=linspace(0,2pi,320); s=linspace(-pi/2,pi/2,160)'; ...
>d=1+0.2*(cos(4*t)+cos(8*s)); ...
>plot3d(cos(t)*cos(s)*d,sin(t)*cos(s)*d,sin(s)*d,hue=1, ...
>  light=[1,0,1],frame=0,zoom=5):
```

Tentu saja, titik awan juga mungkin. Untuk membuat data titik dalam ruang, kami membutuhkan tiga vektor untuk koordiat dari titik-titik.

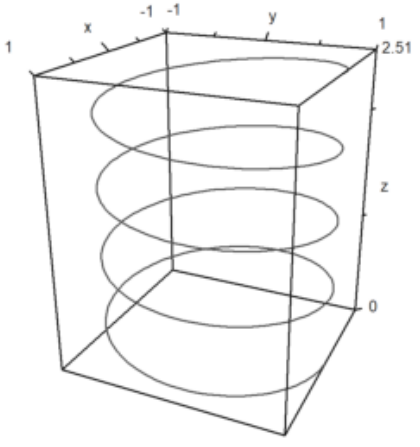
Gaya seperti dalam plot3d dengan points=true;

```
>n=500;  ...
>  plot3d(normal(1,n),normal(1,n),normal(1,n),points=true,style="."):
```

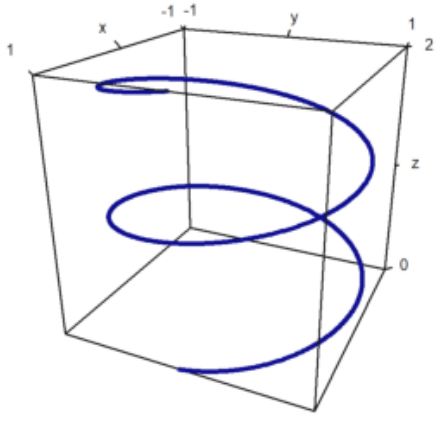


Ini juga mungkin untuk membuat plot sebuah kurva 3D. Dalam kasus ini, mudah untuk menghitung titik-titik dari kurva sebelumnya. Untuk kurva dalam bidang kamu menggunakan urutan dari koordinat dan parameter wire=true.

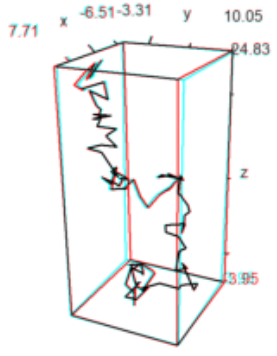
```
>t=linspace(0,8pi,500); ...
>plot3d(sin(t),cos(t),t/10,>wire,zoom=3):
```




```
>t=linspace(0,4pi,1000); plot3d(cos(t),sin(t),t/2pi,>wire, ...
>linewidth=3,wirecolor=blue):
```

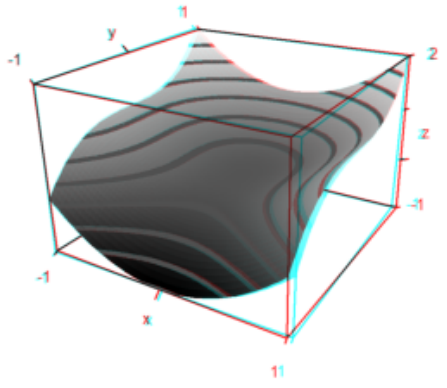



```
>X=cumsum(normal(3,100)); ...
> plot3d(X[1],X[2],X[3],>anaglyph,>wire):
```



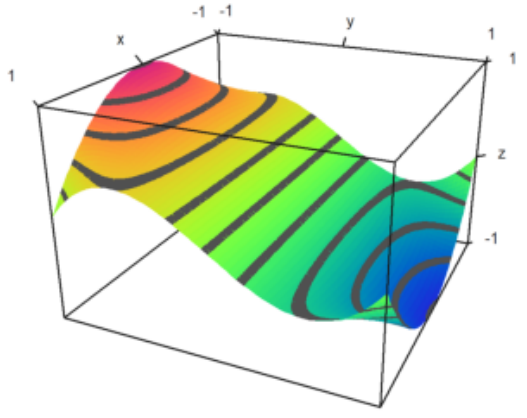
EMT dapat juga membuat plot dalam mode anaglyph. Untuk memandang seperti sebuah plot, kamu membutuhkan kacamata merah/cyan.

```
>plot3d("x^2+y^3",>anaglyph,>contour,angle=30°):
```



Sering kali, sebuah skema spektral warna menggunakan plot. Menegaskan tinggi dari fungsi.

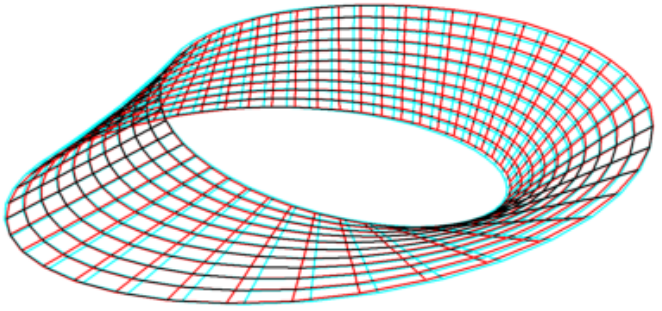
```
>plot3d("x^2*y^3-y",>spectral,>contour,zoom=3.2):
```



Euler dapat membuat permukaan yang terparameter juga, ketika parameter nilai x, y, dan z dari sebuah gambar dari sebuah batasan persegi dalam ruang.

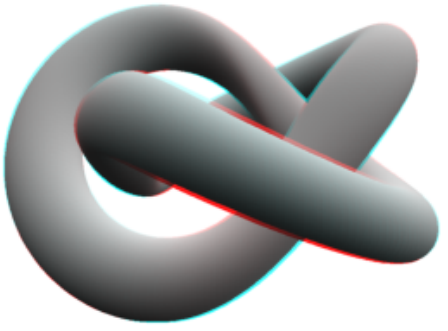
Seperti demo berikut, kami mengatur parameter u dan v, dan membuat koordinat ruang untuk ini.

```
>u=linspace(-1,1,10); v=linspace(0,2*pi,50)'; ...
>X=(3+u*cos(v/2))*cos(v); Y=(3+u*cos(v/2))*sin(v); Z=u*sin(v/2); ...
>plot3d(X,Y,Z,>anaglyph,<frame,>wire,scale=2.3):
```



Ini merupakan contoh yang lebih kompleks, yang mana terlihat megah dengan kacamata merah/cyan.

```
>u:=linspace(-pi,pi,160); v:=linspace(-pi,pi,400)';  ...
>x:=(4*(1+.25*sin(3*v))+cos(u))*cos(2*v); ...
>y:=(4*(1+.25*sin(3*v))+cos(u))*sin(2*v); ...
> z=sin(u)+2*cos(3*v); ...
>plot3d(x,y,z,frame=0,scale=1.5,hue=1,light=[1,0,-1],zoom=2.8,>anaglyph):
```



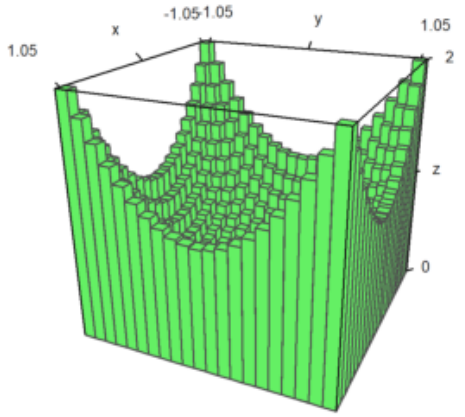
Plot batang juga mungkin untuk dibuat. Seperti ini, kami menyediakan

- x: vektor baris dengan elemen n+1
- y: vektor kolom dengan elemen n+1
- z: matriks nxn nilai.

z dapat diperbesar, tetapi hanya nilai nxn yang akan digunakan.

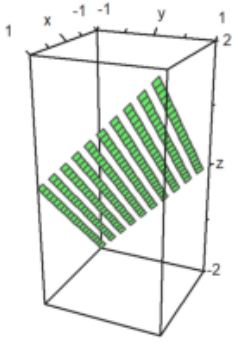
Dalam contoh, kami pertama-tama menghitung nilai. Kemudian kita menyesuaikan x dan y, jadi tengah vektor pada nilai yang digunakan.

```
>x=-1:0.1:1; y=x'; z=x^2+y^2; ...
>xa=(x|1.1)-0.05; ya=(y_1.1)-0.05; ...
>plot3d(xa,ya,z,bar=true):
```



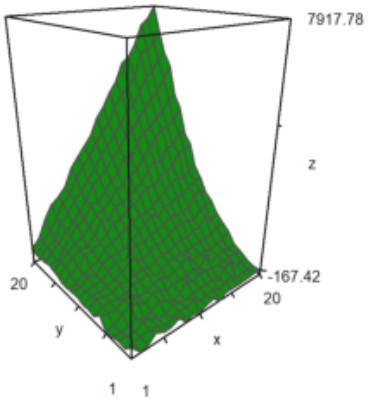
Ini memungkinkan untuk memisah plot dari permukaan dalam dua bagian atau lebih

```
>x=-1:0.1:1; y=x'; z=x+y; d=zeros(size(x)); ...
>plot3d(x,y,z,disconnect=2:2:20):
```

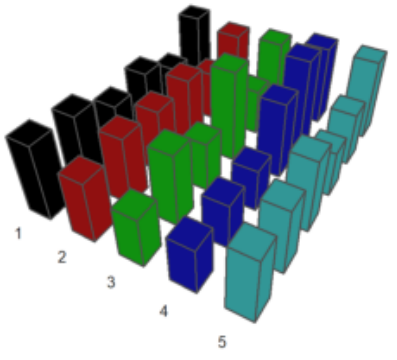


Jika memuat atau menghasilkan sebuah matriks data M dari sebuah file dan membutuhkan untuk membuat plot itu dalam 3D kamu dapat menskalakan matriks ke [-1, 1] dengan skala (M), atau skala matriks dengan >zscale. Ini dapat juga dikombinasikan dengan pemfaktoran skala tersendiri yang aman dapat diaplikasikan sebagai tambahan.

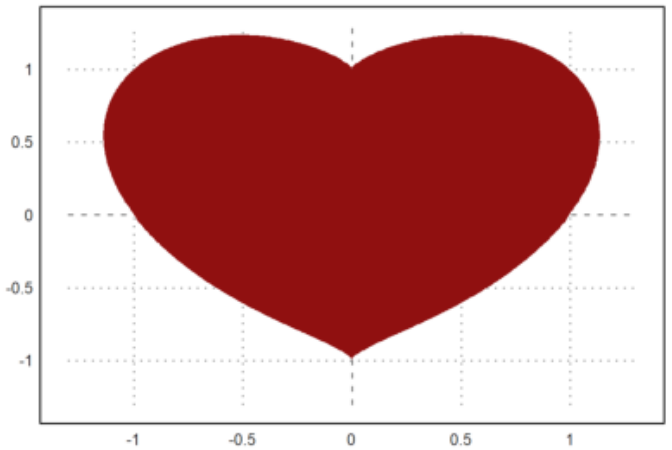
```
>i=1:20; j=i'; ...
>plot3d(i*j^2+100*normal(20,20),>zscale,scale=[1,1,1.5],angle=-40°,zoom=1.8):
```




```
>Z=intrandom(5,100,6); v=zeros(5,6); ...
>loop 1 to 5; v[#]=getmultiplicities(1:6,Z[#]); end; ...
>columnsplot3d(v',scols=1:5,ccols=[1:5]):
```


1
2
3
4
5

```
>plot2d("(x^2+y^2-1)^3-x^2*y^3",r=1.3, ...
>style="#",color=red,<outline, ...
>level=[-2;0],n=100):
```




```
>ekspresi &= (x^2+y^2-1)^3-x^2*y^3; $ekspresi
```

$$\left(y^2+x^2-1\right)^3-x^2\,y^3$$

Kami mengharapkan untuk merubah kurva hati secara mengelilingi sumbu-y. Ini adalah ekspresi, yang mana mendefinisikan hati:

$$f(x,y) = (x^2+y^2-1)^3 - x^2.y^3.$$

Selanjutnya kami mengatur

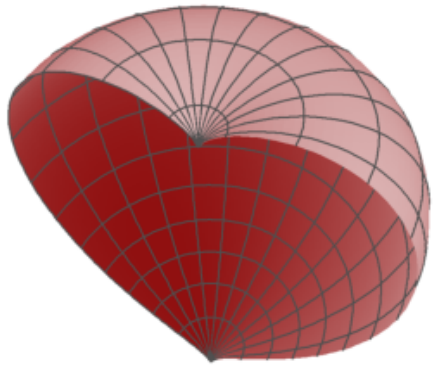
$$x = r.cos(a), \quad y = r.sin(a).$$

```
>function fr(r,a) &= ekspresi with [x=r*cos(a),y=r*sin(a)] | trigreduce; $fr(r,a)
```

$$\left(r^2-1\right)^3 + \frac{\left(\sin\left(5\,a\right) - \sin\left(3\,a\right) - 2\,\sin a\right)\,r^5}{16}$$

Ini memperbolehkan untuk mendefinisikan fungsi numerik, yang mana menyelesaikan untuk r, jika diberikan a. Dengan fungsi itu kami dapat membuat plot dengan mengubah hati sebagai permukaan parametrik

```
>function map f(a) := bisect("fr",0,2;a); ...
>t=linspace(-pi/2,pi/2,100); r=f(t);  ...
>s=linspace(pi,2pi,100)'; ...
>plot3d(r*cos(t)*sin(s),r*cos(t)*cos(s),r*sin(t), ...
>>hue,<frame,color=red,zoom=4,amb=0,max=0.7,grid=12,height=50°):
```

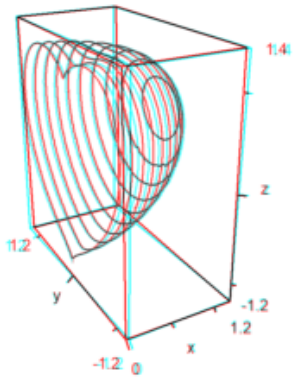


Berikut merupakan plot 3D dari penggambaran diatas diputar mengelilingi sumbu-y. Kami mendefinisikan fungsi yang mana mendeskripsikan objek

```
>function f(x,y,z) ...
```

```
  r=x^2+y^2;
  return (r+z^2-1)^3-r*z^3;
   endfunction
```

```
>plot3d("f(x,y,z)", ...
>xmin=0,xmax=1.2,ymin=-1.2,ymax=1.2,zmin=-1.2,zmax=1.4, ...
>implicit=1,angle=-30°,zoom=2.5,n=[10,100,60],>anaglyph):
```


1.4
z
1.2
y
-1.2
-1.2
0
x
1.2

Plot 3D Spesial

Fungsi plot3d sangat bagus untuk dimiliki, tetapi tidak memenuhi semua yang kita butuhkan. Disamping rutinitas umum yang lainnya, ini mungkin untuk mendapatkan sebuah plot berframe dari sebarang objek yang kamu sukai.

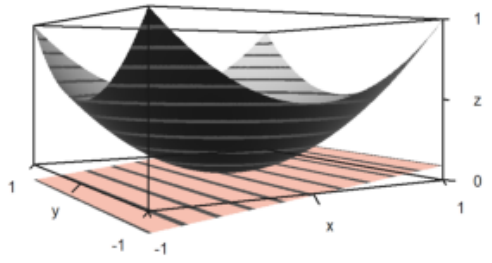
Meskipun Euler bukanlah program 3D, ini dapat mengkombinasikan objek objek sederhana. Kami mencoba untuk memvisualisasikan sebuah paraboloida dan garis singgung.

```
>function myplot ...
```

```
    y=-1:0.01:1; x=(-1:0.01:1)';
    plot3d(x,y,0.2*(x-0.1)/2,<scale,<frame,>hue, ..
      hues=0.5,>contour,color=orange);
    h=holding(1);
    plot3d(x,y,(x^2+y^2)/2,<scale,<frame,>contour,>hue);
    holding(h);
  endfunction
```

Sekarang framedplot() menyediakan frame dan mengatur pandangan.

```
>framedplot("myplot",[-1,1,-1,1,0,1],height=0,angle=-30°, ...
>  center=[0,0,-0.7],zoom=3):
```

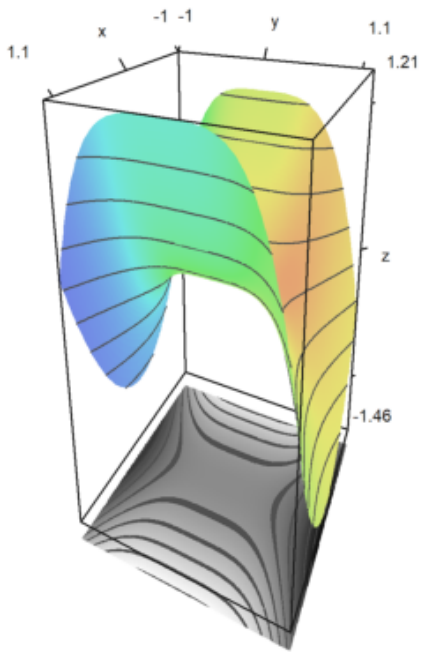


Dengan jalan yang sama, kamu dapat mem-plotkan bidang kontur secara manual. Catatan bahwa plot3d() secara bawaaan mengatur jendela ke fullwindow(), tetapi asumsikan demikian untuk plotcontourplane().

```
>x=-1:0.02:1.1; y=x'; z=x^2-y^4;
>function myplot (x,y,z) ...
```

```
    zoom(2);
    wi=fullwindow();
    plotcontourplane(x,y,z,level="auto",<scale);
    plot3d(x,y,z,>hue,<scale,>add,color=white,level="thin");
    window(wi);
    reset();
  endfunction
```

```
>myplot(x,y,z):
```



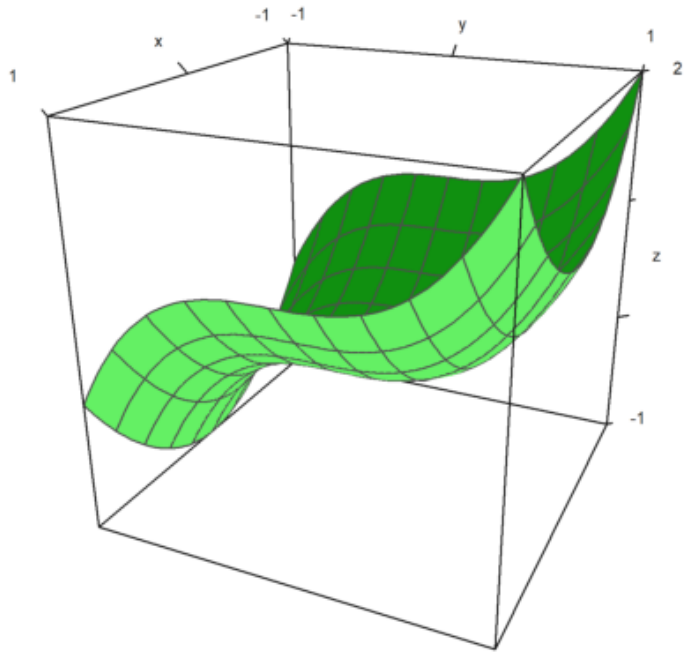
Animasi

Euler dapat menggunakan frame untuk menghitung animasi dalam langkah sebelumya

Satu fungsi, yang mana membuat teknik ini digunakan adalah rotate. Ini dapat merubah sudut dari pandang dan menggambar ulang sebuah plot 3D. Pemanggulan fungsi addpage() untuk setiap plot baru. Akhirnya plot ini dianimasikan.

Tolong pelajari sumber dari rotasi untuk lihat lebih detail.

```
>function testplot () := plot3d("x^2+y^3"); ...
>rotate("testplot"); testplot():
```


-1
-1
x
y
1
1
2
z
-1

Menggambar Povray

Dengan bantuan dari file Euler povray.e, Euler dapat membuat file Povray. Hasilnya sangat bagus untuk dilihat.

Kamu butuh menginstall Povray (32bit atau 64bit) dari http://www.povray.org/, dan ambillah subdirektori "bin" dari Povray ke environtment path atau atur ke variabel "defaultpovray" dengan path yang lengkap untuk menunjukkan ke "pvengine.exe".

Tampilan Povray dari Euler membuat file Povray dalam direktori home dari pengguna, dan memanggil Povray untuk menguraikan file ini. Nama file bawaan adalah current.pov dan direktori bawaan adalah eulerhome(), biasanya c:\Users\Username\Euler. Povray membuat sebuah file PNG, yang mana dimuat oleh Euler kedalam sebuah notebook. Untuk membersihkan file ini, gunakan povclear().

Fungsi pov3d sama seperti plot3d. Ini akan menghasilkan grafik dari fungsi f(x, y) atau sebuah permukaan dengan koordinat X, Y, Z dalam matriks, termasuk secara opsional garis level. Fungsi ini dimulai secara otomatis raytracer dan memuat pemandangan kedalam Euler notebook.

Disamping pov3d(), terdapat banyak sekali fungsi, yang mana menghasilkan objek Povray. Fungsi ini mengembalikan strings, mencakup kode Povray untuk objek. Untuk menggunakan fungsi-fungsi ini, mulai file Povray dengan povstart(). Lalu gunakan writeln(...) untuk menulis objek ke dalam file pandangan. Terakhir, akhiri file dengan povend(). Secara bawaan, raytracer akan mulai dan file PNG akan dimasukkan kedalam notebook Euler.

Fungsi objek memiliki sebuah parameter yang bernama "look", yang mana membutuhkan string dengan kode Povray untuk tekstur dan akhiri objek. Fungsi povlook() dapat digunakan untuk menghasilkan string ini. Ini memiliki parameter untuk warna, transparansi, bayangan Phong, dan lainnya

Catatan bahwa semesta Povray memiliki sistem koordinat lain. Antar muka ini menerjemahkan semua koordinat kedalam sistem Povray. Jadi kamu dapat tetap berpikir dalam koordinat sistem Euler dengan menunjukkan z secara vertikal keatas, dan sumbu x, y, z dalam penalaran sistem tangan kanan.

Kamu butuh untuk memuat file povray.

```
>load povray;
```

Pastikan, direktori bin Povray dalam jalur. Jika tidak atur mengikuti variabel, jadi ini mengandung jalur ke executable povray.

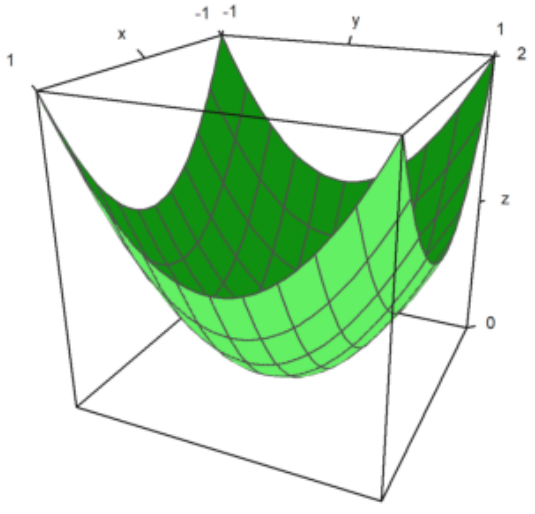
```
>defaultpovray="C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe"
```

```
C:\Program Files\POV-Ray\v3.7\bin\pvengine.exe
```

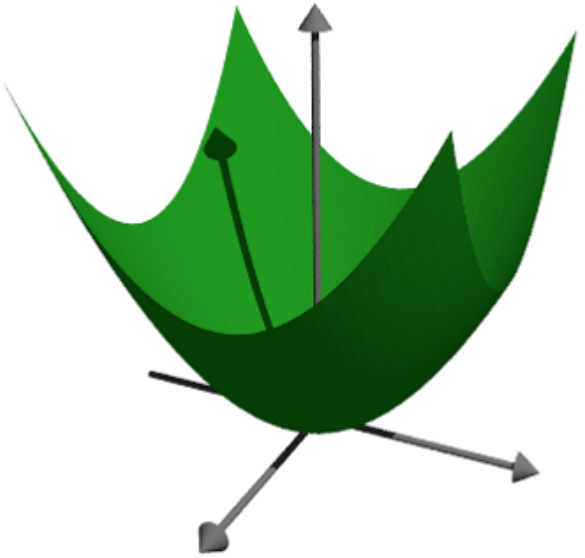
Untuk pertama kali pemakaian, kami mem-plotkan fungsi sederhana. Perintah berikut membuat sebuah file povray dalam direktori pengguna kamu, dan menjalankan Povray untuk penelusuran sinar dalam file.

Jika kamu memulai perintah berikut, GUI Povray seharusnya terbuka, menjalankan file, dan menutup secara otomatis. Karena alasan keamanan, kamu akan diminta, untuk memberikan izin untuk menjalankan file exe. Kamu dapat mengklik cancel untuk memberhentikan pertanyaan selanjutnya. Kamu mungkin akan mengklik OK dalam jendela Povray untuk menyetujui percakapan awal dari Povray.

```
>plot3d("x^2+y^2",zoom=2):
```

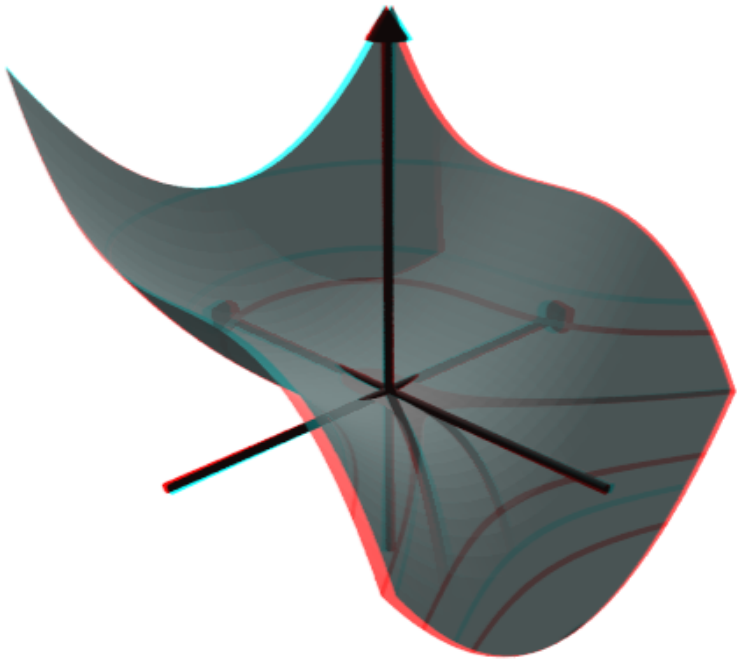



```
>pov3d("x^2+y^2",zoom=3);
```



Kami dapat membuat fungsi transparan dan menambahkan pada hasil akhir lainnya. Kami juga dapat menambahkan garis level ke fungsi plot.

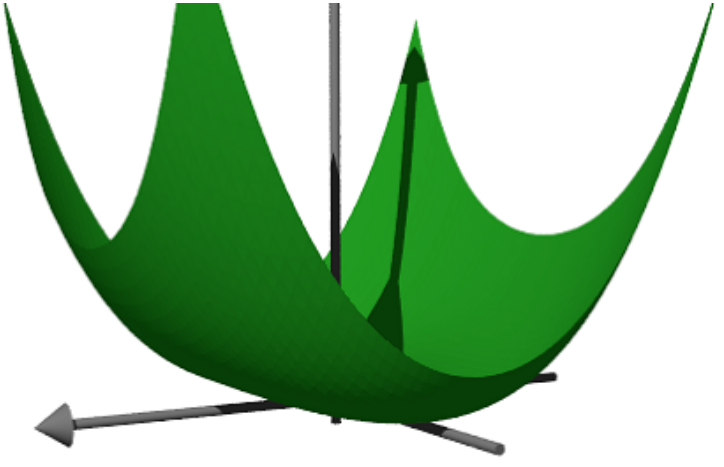
```
>pov3d("x^2+y^3",axiscolor=red,angle=-45°,>anaglyph, ...
>  look=povlook(cyan,0.2),level=-1:0.5:1,zoom=3.8);
```



Terkadang ini dibutuhkan untuk mencegah penskalaan fungsi dan skala fungsi dengan tangan.

Kami memplot himpunan titik-titik dalam bidang kompleks, dimana hasil dari jarak ke 1 dan -1 sama dengan 1.

```
>pov3d("((x-1)^2+y^2)*((x+1)^2+y^2)/40",r=2, ...
>  angle=-120°,level=1/40,dlevel=0.005,light=[-1,1,1],height=10°,n=50, ...
>  <fscale,zoom=3.8);
```

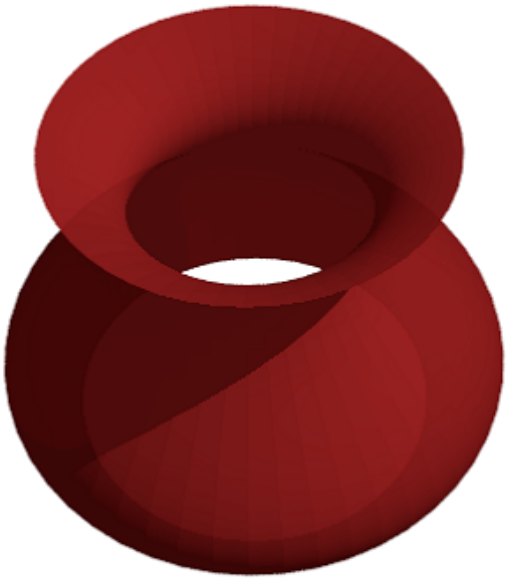


Membuat Plot dengan Koordinat

Selain fungsi, kami dapat mem-plot dengan koordinat. Seperti dalam plot3d, kami membutuhkan tiga matiks untuk mendefinisikan objek.

Dalam contoh kami memutar mengelilingi sumbu-z.

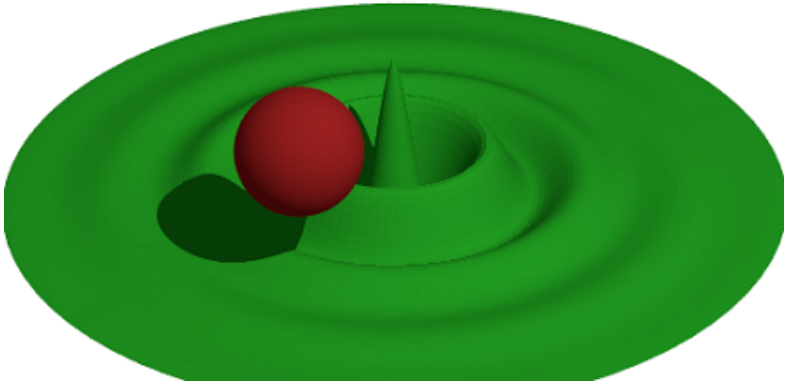
```
>function f(x) := x^3-x+1; ...
>x=-1:0.01:1; t=linspace(0,2pi,50)'; ...
>Z=x; X=cos(t)*f(x); Y=sin(t)*f(x); ...
>pov3d(X,Y,Z,angle=40°,look=povlook(red,0.1),height=50°,axis=0,zoom=4,light=[10,5,15]);
```



Dalam contoh berikut, kami membuat plot glombang yang mengecil. Kami membuat gelombang dengan bahasa matriks Euler.

Kami juga menunjukkan tambahan objek yang dapat ditambahkan ke layar pov3d. Untuk pembuatan objek, lihat contoh berikut. Catatan bahwa plot3d menskalakan plot, jadi ini akan menyesuaikan ke unit kubus.

```
>r=linspace(0,1,80); phi=linspace(0,2pi,80)'; ...
>x=r*cos(phi); y=r*sin(phi); z=exp(-5*r)*cos(8*pi*r)/3;  ...
>pov3d(x,y,z,zoom=6,axis=0,height=30°,add=povsphere([0.5,0,0.25],0.15,povlook(red)), ...
>  w=500,h=300);
```



Dengan metode bayangan lebih lanjut dari Povray, sangat sedikit titik yang dapat menghasilkan permukaan yang sangat halus. Hanya pada saat batasan dan dalam bayangan trik ini mungkin akan terlihat jelas.

Untuk ini, kami membutuhkan vektor normal dalam setiap titik matriks.

```
>Z &= x^2*y^3
```

$$x^2\,y^3$$

Persamaan dari permukaan adalah [x, y, Z]. Kami menghitung dua turunan ke x dan y dari ini dan mengambil hasil kali silang sebagai normal.

```
>dx &= diff([x,y,Z],x); dy &= diff([x,y,Z],y);
```

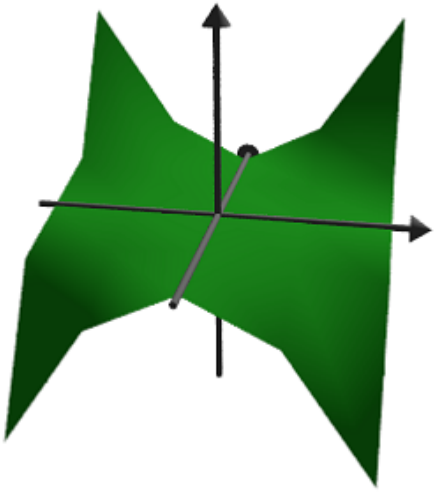
Kami mendefinisikan normal sebagai perkalian produk dari turunan ini dan mendefinisikan fungsi koordinat.

```
>N &= crossproduct(dx,dy); NX &= N[1]; NY &= N[2]; NZ &= N[3]; N,
```

$$\left[-2\,x\,y^3,\ -3\,x^2\,y^2,\ 1\right]$$

Kami hanya menggunakan 25 titik.

```
>x=-1:0.5:1; y=x';
>pov3d(x,y,Z(x,y),angle=10°, ...
>  xv=NX(x,y),yv=NY(x,y),zv=NZ(x,y),<shadow);
```



Berikut merupakan knot Trefoil dilakukan dengan A. Busser dalam Povray. Ini merupakan versi yang telah diperbarui dalam contoh ini.

Contoh: Examples\Trefoil Knot | Trefoil Knot

Untuk pemandangan yang bags dengan tidak beberapa titik, kami menambahkan vektor normal disini. KAmi menggunakan Maxima untuk menghitung normal untuk kami. Pertama, tiga fungsi untuk koordinat sebagai ekspresi simbolik.

```
>X &= ((4+sin(3*y))+cos(x))*cos(2*y); ...
>Y &= ((4+sin(3*y))+cos(x))*sin(2*y); ...
>Z &= sin(x)+2*cos(3*y);
```

Lalu dua vektor turunan ke x dan y.

```
>dx &= diff([X,Y,Z],x); dy &= diff([X,Y,Z],y);
```

Sekarang normal, yang mana adalah hasil kali dari dua turunan.

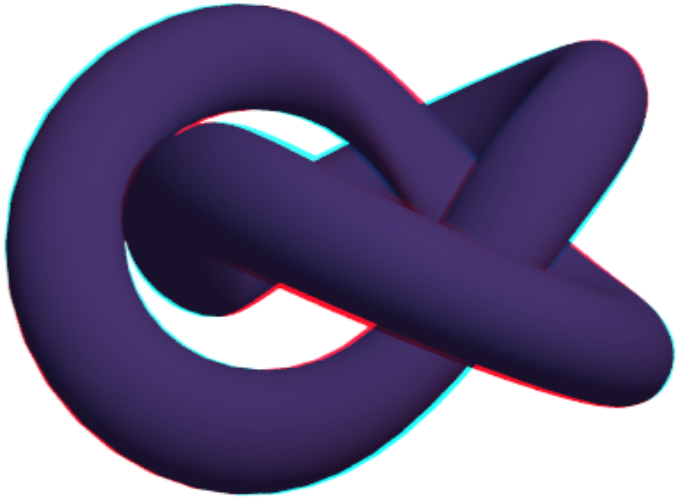
```
>dn &= crossproduct(dx,dy);
```

Kami sekarang menilai semua ini secara numerik.

```
>x:=linspace(-%pi,%pi,40); y:=linspace(-%pi,%pi,100)';
```

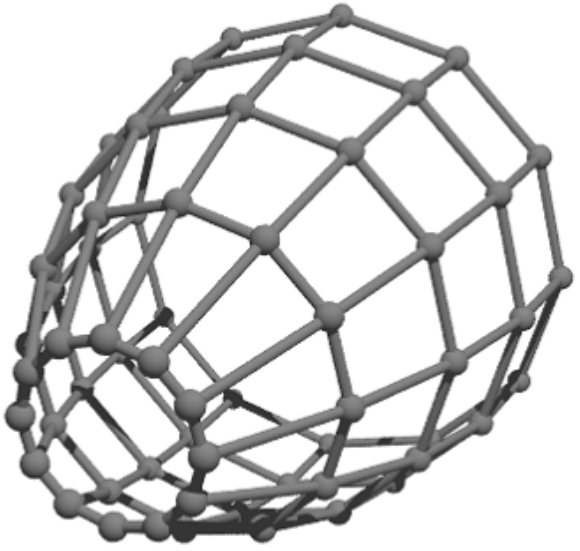
Vektor-vektor normal merupakan penilaian dari ekspresi simbolik dn[i] untuk i=1,2,3. Sintaks untuk ini adalah &"ekspresi"(parameter). Ini merupakan sebuah alternatif dari metode dalam contoh sebelumnya, kami dapat mendefinisikan ekspresi simbolic NX, NY, NZ terlebih dahulu.

```
>pov3d(X(x,y),Y(x,y),Z(x,y),>anaglyph,axis=0,zoom=5,w=450,h=350, ...
>  <shadow,look=povlook(blue), ...
>  xv=&"dn[1]"(x,y), yv=&"dn[2]"(x,y), zv=&"dn[3]"(x,y));
```



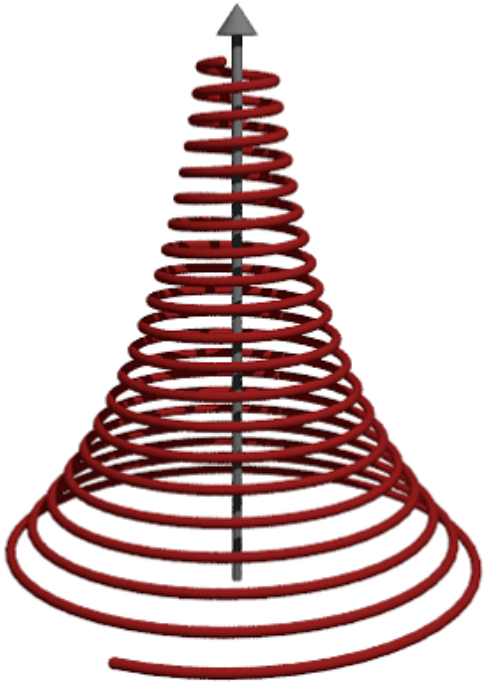
Kami juga dapat menghasilkan jaring-jaring dalam 3D.

```
>povstart(zoom=4); ...
>x=-1:0.5:1; r=1-(x+1)^2/6; ...
>t=(0°:30°:360°)'; y=r*cos(t); z=r*sin(t); ...
>writeln(povgrid(x,y,z,d=0.02,dballs=0.05)); ...
>povend();
```



Dengan povgrid(), kurva-kurva jadi memungkinkan.

```
>povstart(center=[0,0,1],zoom=3.6); ...
>t=linspace(0,2,1000); r=exp(-t); ...
>x=cos(2*pi*10*t)*r; y=sin(2*pi*10*t)*r; z=t; ...
>writeln(povgrid(x,y,z,povlook(red))); ...
>writeAxis(0,2,axis=3); ...
>povend();
```



Objek Povray

Diatas, kami menggunakan pov3d untuk mem-plot permukaan. Antarmka povray dalam Euler dapat membuat objek Povray. Objek-objek ini disimpan sebagai string dalam Euler, dan perlu untuk menulis ke file Povray.

Kami mulai mengeluarkan dengan povstart().

```
>povstart(zoom=4);
```

Pertama kami definisikan tiga silinder dan menyimpanya ke string dalam Euler.

Fungsi-fungsi povx() dan lainnya secara sederhana mengembalikan vektor [1, 0, 0], yang mana dapat digunakan sebagai gantinya.

```
>c1=povcylinder(-povx,povx,1,povlook(red)); ...
>c2=povcylinder(-povy,povy,1,povlook(yellow)); ...
>c3=povcylinder(-povz,povz,1,povlook(blue)); ...
```

The strings contain Povray code, which we need not understand at that point.

```
>c2
```

```
cylinder { <0,0,-1>, <0,0,1>, 1
 texture { pigment { color rgb <0.941176,0.941176,0.392157> }  }
 finish { ambient 0.2 }
 }
```

Seperti yang kamu lihat, kami menammbahkan tekstur ke objek dalam tiga warna yang berbeda.

Ini dapat dilakukan dengan povlook(), yang mana mengembalikan sebuah stirng dengan code Povray yang relevan. Kami dapat menggunakan warna bawaan Euler, atau mendefinisikan sendiri warna kami. Kami jua dapat menambahkan transparansi, atau mengubah cahaya sekitar.

```
>povlook(rgb(0.1,0.2,0.3),0.1,0.5)
```

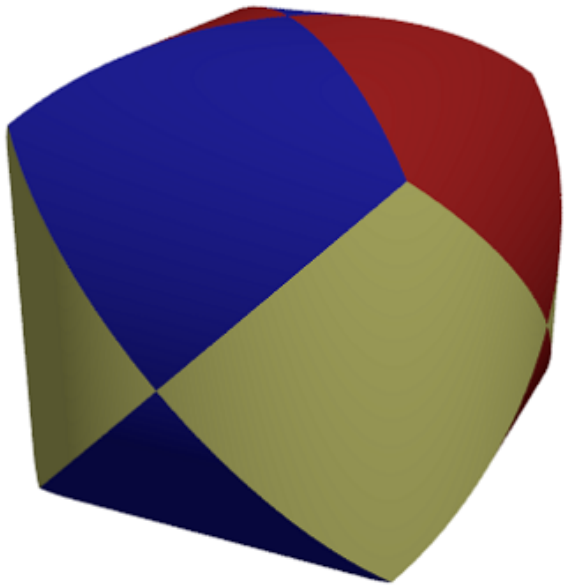
```
  texture { pigment { color rgbf <0.101961,0.2,0.301961,0.1> }  }
  finish { ambient 0.5 }
```

Sekarang kami mendefinisikan objek yang saling memotong dan menulis hasilnya kedalam file.

```
>writeln(povintersection([c1,c2,c3]));
```

Potongan dari silinder tiga sangat sulit untuk digambarkan, jika kamu belum pernah melihatnya.

```
>povend;
```



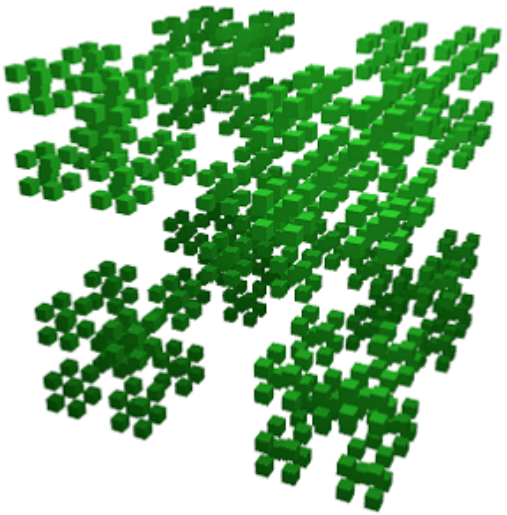
Fungsi-fungsi berikut membuat sebuah fraktal secara rekursif.

Fungsi pertama menunjukkan, bagaimana Euler menangani objek Povray sederhana. povbox() fungsi mengembalikan sebuah string, yang mana memuat koordinat-koordinat boks, tekstur dan selesai.

```
>function onebox(x,y,z,d) := povbox([x,y,z],[x+d,y+d,z+d],povlook());
>function fractal (x,y,z,h,n) ...
```

```
  if n==1 then writeln(onebox(x,y,z,h));
  else
    h=h/3;
    fractal(x,y,z,h,n-1);
    fractal(x+2*h,y,z,h,n-1);
    fractal(x,y+2*h,z,h,n-1);
    fractal(x,y,z+2*h,h,n-1);
    fractal(x+2*h,y+2*h,z,h,n-1);
    fractal(x+2*h,y,z+2*h,h,n-1);
    fractal(x,y+2*h,z+2*h,h,n-1);
    fractal(x+2*h,y+2*h,z+2*h,h,n-1);
    fractal(x+h,y+h,z+h,h,n-1);
  endif;
endfunction
```

```
>povstart(fade=10,<shadow);
>fractal(-1,-1,-1,2,4);
>povend();
```



Perbedaan memungkinkan pemotongan satu objek dari objek lainnya. Seperti perpotongan, ini merupakan bagian dari objek CSG dari Povray.

```
>povstart(light=[5,-5,5],fade=10);
```

Untuk demonstrasi ini, kami mendefinisikan sebuah objek dalam Povray, daripada menggunakan sebuah string dalam Euler. Definisi ditulis ke file secara langsung.

Sebuah koordinat dimensi tiga dari -1 berarti [-1, -1, -1].

```
>povdefine("mycube",povbox(-1,1));
```

Kami menggunakan objek ini dengan povobject(), yang mana mengembalikan sebuah string seperti biasanya.

```
>c1=povobject("mycube",povlook(red));
```

Kami membuat sebuah kubus kedua dan merotasi dan men-skalakan sedikit.

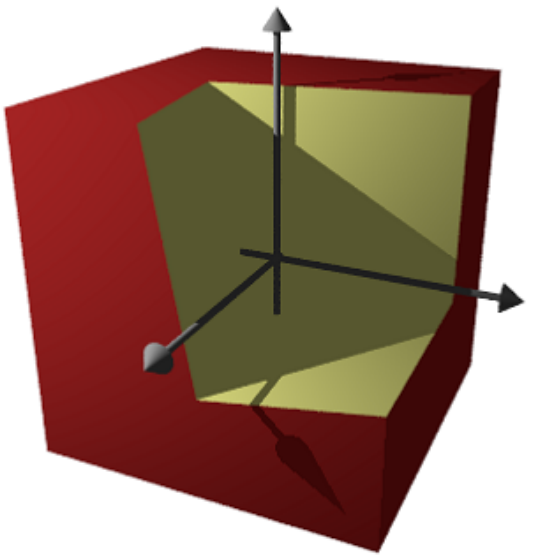
```
>c2=povobject("mycube",povlook(yellow),translate=[1,1,1], ...
>  rotate=xrotate(10°)+yrotate(10°), scale=1.2);
```

Lalu kami ambil selisih kedua objek tersebut.

```
>writeln(povdifference(c1,c2));
```

Sekarang tambahkan tiga sumbu.

```
>writeAxis(-1.2,1.2,axis=1); ...
>writeAxis(-1.2,1.2,axis=2); ...
>writeAxis(-1.2,1.2,axis=4); ...
>povend();
```



Fungsi-Fungsi Implisit

Povray dapat mem-plot himpunan yang mana f(x, y, z) = 0, seperti parameter implisit dalam plot3d. Namun, hasilnya terlihat lebih baik.

Sintaks untuk fungsi-fungsi sedikit berbeda. Kamu tidak dapat menggunakan keluaran dari Maxima atau ekspresi Euler.

$$((x^2+y^2-c^2)^2+(z^2-1)^2)*((y^2+z^2-c^2)^2+(x^2-1)^2)*((z^2+x^2-c^2)^2+(y^2-1)^2)=d$$

```
>povstart(angle=70°,height=50°,zoom=4);
>c=0.1; d=0.1; ...
>writeln(povsurface("(pow(pow(x,2)+pow(y,2)-pow(c,2),2)+pow(pow(z,2)-1,2))*(pow(pow(y,2)+pow(z,2)-po
>povend();
```

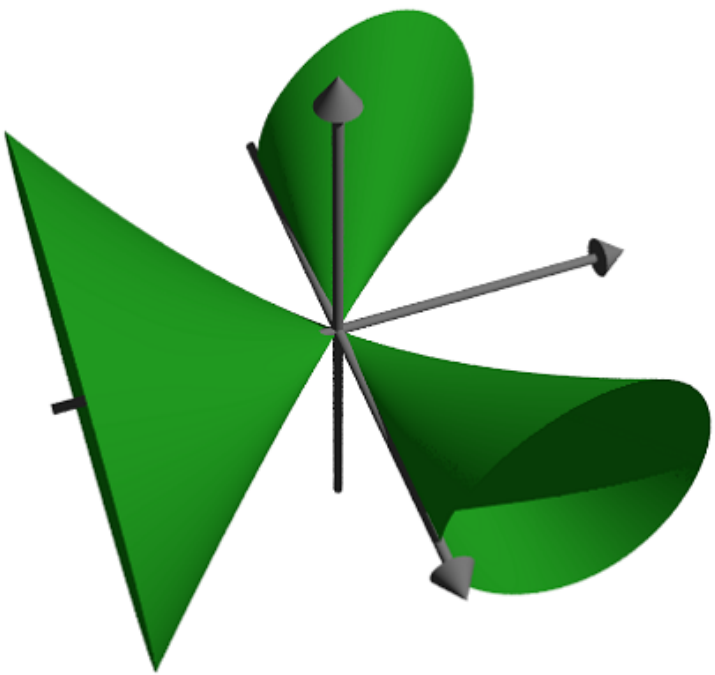
```
Error : Povray error!

Error generated by error() command

povray:
    error("Povray error!");
Try "trace errors" to inspect local variables after errors.
povend:
    povray(file,w,h,aspect,exit);
```

```
>povstart(angle=25°,height=10°);
>writeln(povsurface("pow(x,2)+pow(y,2)*pow(z,2)-1",povlook(blue),povbox(-2,2,"")));
>povend();
```

```
>povstart(angle=70°,height=50°,zoom=4);
```

Buat permukaan implisit. Catat sintaks yang berdeda dari ekspresi.

```
>writeln(povsurface("pow(x,2)*y-pow(y,3)-pow(z,2)",povlook(green))); ...
>writeAxes(); ...
>povend();
```



Objek Mesh

Contoh ini, kami menunjukkan bagaimana untuk membuat objek mesh, dan menggambarnya dengan informasi tambahan.

Kami ingin memaksimalkan xt dibawah kondisi x + y = 1 dan mendemonstrasikan sentuan tangential dari garis level.

```
>povstart(angle=-10°,center=[0.5,0.5,0.5],zoom=7);
```

Kami tidak dapat menyimpan objek dalam string seperti sebelumnya, karena terlalu besar. Jadi kami mendefinisikan objek dalam file Povray menggunakan declare. Fungsi povtriangle() melakukan ini secara otomatis. Itu dapat menerima vektor normal seperti pov3d().

Dibawah ini mendefinisikan objek mesh dan menulisnya langsung kedalam file.

```
>x=0:0.02:1; y=x'; z=x*y; vx=-y; vy=-x; vz=1;
>mesh=povtriangles(x,y,z,"",vx,vy,vz);
```

Sekarang kami mendefinisikan dua cakram, yang mana akan dipotong dengan permukaan.

```
>cl=povdisc([0.5,0.5,0],[1,1,0],2); ...
>ll=povdisc([0,0,1/4],[0,0,1],2);
```

Tulis pemukaan minus dua cakram.

```
>writeln(povdifference(mesh,povunion([cl,ll]),povlook(green)));
```

Tulis potongan keduanya.

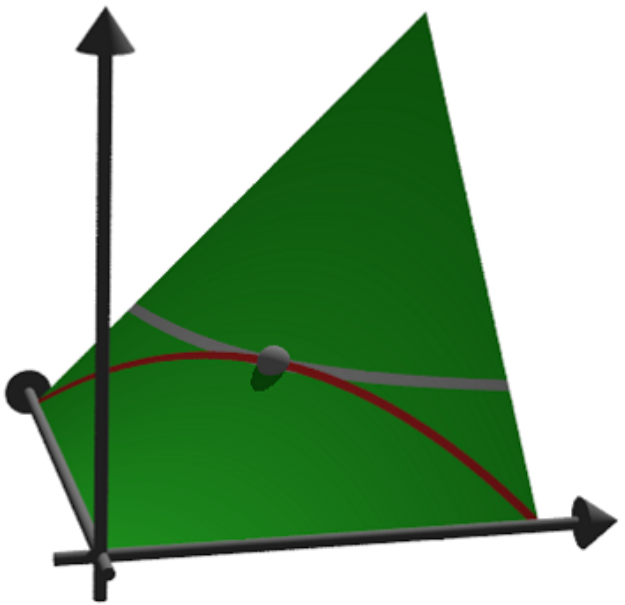
```
>writeln(povintersection([mesh,cl],povlook(red)); ...
>writeln(povintersection([mesh,ll],povlook(gray)));
```

Tulis maksimal titik.

```
>writeln(povpoint([1/2,1/2,1/4],povlook(gray),size=2*defaultpointsize));
```

Tambahkan sumbu dan selesai.

```
>writeAxes(0,1,0,1,0,1,d=0.015); ...
>povend();
```



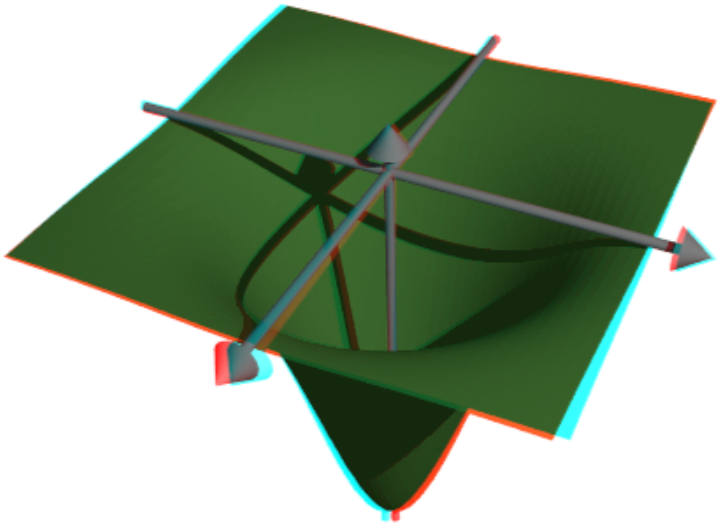
Anaglyphs dalam Povray

Untuk membuat sebuah anaglyph untuk kacamata merah/cyan, Povray harus menjalankan dua kamera dengan posisi yang berbeda. Ini akan menghasilkan dua Povray file dan dua file PNG, yang mana dimuat dalam fungsi loadanaglyph().

Tentu saja, kamu membutuhkan kacamata merah/cyan untuk melihat contoh berikut secara benar.

Fungsi pov3d() memiliki penggantian sederhana untuk membuat anaglyphs.

```
>pov3d("-exp(-x^2-y^2)/2",r=2,height=45°,>anaglyph, ...
>  center=[0,0,0.5],zoom=3.5);
```



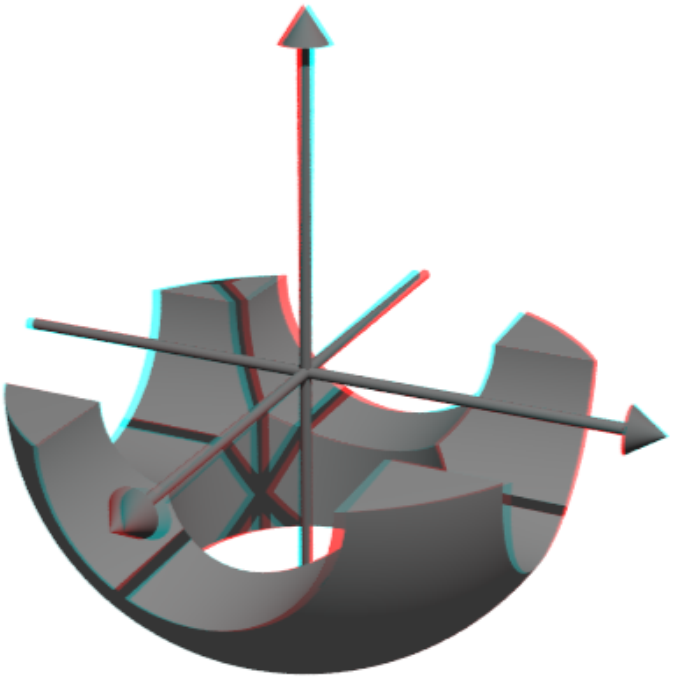
Jika kamu membuat sebuah pemandangan dengan objek, kamu perlu untuk menempatkan pembuatan pemandangan ke dalama suatu fungsi dan jalankan keduannya dengan nilai berbeda untuk parameter anaglyph.

```
>function myscene ...
```

```
    s=povsphere(povc,1);
    cl=povcylinder(-povz,povz,0.5);
    clx=povobject(cl,rotate=xrotate(90°));
    cly=povobject(cl,rotate=yrotate(90°));
    c=povbox([-1,-1,0],1);
    un=povunion([cl,clx,cly,c]);
    obj=povdifference(s,un,povlook(red));
    writeln(obj);
    writeAxes();
  endfunction
```

Fungsi povanglyph() akan menjalankan ini semua. Parameter seperti povstart() dan povend() dikombinasikan.

```
>povanaglyph("myscene",zoom=4.5);
```



Mendefinisikan Objek milik sendiri

Tampilan povray dari Euler memiliki banyak objek. Tetapi kamu tidak dibatasi olehnya. Kamu dapat membuat objekmu tersendiri, yang mana mengombinasikan objek yang lainnya, atau secara penuh objek baru.

Kami mendemonstrasikan sebuah torys. Perintah Povray untuk ini adalah "torus". Jadi kami mengembalikan sebuah string dengan perintah ini dan parameternya. Catatan bahwa torus selalu berada ditengan tepat pada titik asal.

```
>function povdonat (r1,r2,look="") ...
```

```
    return "torus {"+r1+","+r2+look+"}";
  endfunction
```

Ini merupakan torus pertama kami.

```
>t1=povdonat(0.8,0.2)
```

```
  torus {0.8,0.2}
```

Andaikan kami menggunakan objek ini untuk membuat sebuah torus kedua, mentranslatikan dan merotasinya.

```
>t2=povobject(t1,rotate=xrotate(90°),translate=[0.8,0,0])
```

```
object { torus {0.8,0.2}
 rotate 90 *x
 translate <0.8,0,0>
 }
```

Sekarang kita menempatkan objek kedalam sebuah tempat. Untuk tampilannya, kami menggunakan Phong Shading.

```
>povstart(center=[0.4,0,0],angle=0°,zoom=3.8,aspect=1.5); ...
>writeln(povobject(t1,povlook(green,phong=1))); ...
>writeln(povobject(t2,povlook(green,phong=1))); ...
```

```
>povend();
```

memanggil program Povray. Namun, dalam kasus error, ini tidak akan menampilkan error. Kamu harus melakukan

```
>povend(<exit);
```

jika apapun tidak bekerja. Ini akan meniggalkan jendela Povray terbuka.

```
>povend(h=320,w=480);
```

Ini merupakan contoh yang lebih elaborasi. Kami menyelesaikan

$$Ax \leq b, \quad x \geq 0, \quad c.x \to \text{Max}.$$

dan melihatkan titik-titik yang mungkin dan optimal dalam plot 3D.

```
>A=[10,8,4;5,6,8;6,3,2;9,5,6];
>b=[10,10,10,10]';
>c=[1,1,1];
```

Pertama, mari kita cek, jika contoh ini memiliki sebuah solusi pada semuanya.

```
>x=simplex(A,b,c,>max,>check)'
```

```
  [0,  1,  0.5]
```

Ya, itu mempunyainya.

Selanjutnya kami mendefinisikan dua objek. Pertama merupakan sebuah bidang

$$a \cdot x \leq b$$

```
>function oneplane (a,b,look="") ...
```

```
    return povplane(a,b,look)
  endfunction
```

Lalu kami mendefinisikan perpotongan dari pertengahan bidang semua dan sebuah kubus.

```
>function adm (A, b, r, look="") ...
```

```
    ol=[];
    loop 1 to rows(A); ol=ol|oneplane(A[#],b[#]); end;
    ol=ol|povbox([0,0,0],[r,r,r]);
    return povintersection(ol,look);
  endfunction
```

Sekarang kami dapat membuat plot hasilnya.

```
>povstart(angle=120°,center=[0.5,0.5,0.5],zoom=3.5); ...
>writeln(adm(A,b,2,povlook(green,0.4))); ...
>writeAxes(0,1.3,0,1.6,0,1.5); ...
```

Berikut merupakan sebuah lingkaran di sekitar optimum.

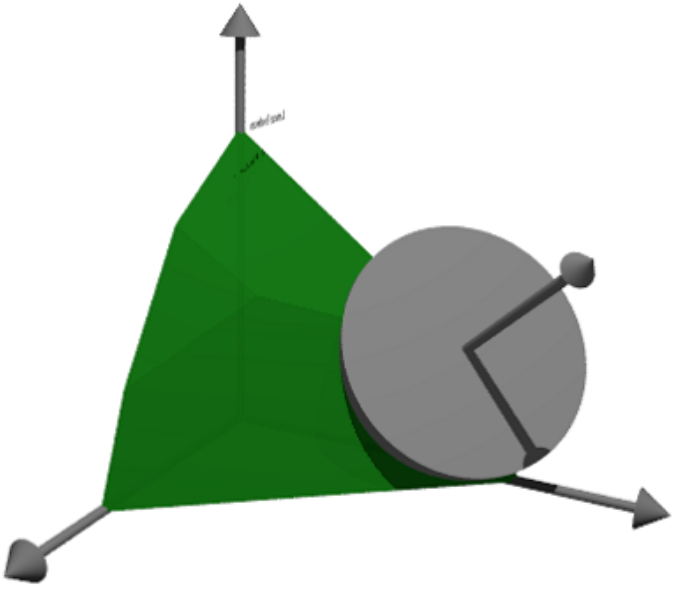
```
>writeln(povintersection([povsphere(x,0.5),povplane(c,c.x')], ...
>  povlook(red,0.9)));
```

Dan sebuah error dalam arah dari optimum.

```
>writeln(povarrow(x,c*0.5,povlook(red)));
```

Kami tambahkan teks ke layar. Teks hanya sebuah objek 3D. Kami butuh tempat untuk membalikkannya mengikuti pandangan kita.

```
>writeln(povtext("Linear Problem",[0,0.2,1.3],size=0.05,rotate=5°)); ...
>povend();
```



Contoh Lainnya

Kamu dapat menemukan contoh lainnya untuk Povray dalam Euler dengan file berikut.

See: Examples/Dandelin Spheres
See: Examples/Donat Math
See: Examples/Trefoil Knot
See: Examples/Optimization by Affine Scaling