

```

/*=====*\
filename: Dandelin.mu
-----\
scopo: routine per disegnare le sfere di Dandelin
relative alla conica identificata dall'in-
tersezione del piano  $ax + by + cz + d = 0$ 
con il doppio cono avente angolo di aper-
tura  $\alpha$ , asse coincidente con l'asse  $z$ ,
vertice in  $O(0,0,0)$ .
-----\
autore: Claudio Marsan
Liceo cantonale di Mendrisio
Via Agostino Maspoli
CH-6850 Mendrisio
claudio.marsan@liceomendrisio.ch
-----\
ultima modifica: 18.12.2005
-----\
testato con: MuPAD Pro 3.1.1 for Windows
-----\
OS: Microsoft Windows XP Professional sp2
-----\
uso: read("../\Dandelin.mu"):
Dandelin(a, b, c, d, alpha)
/*=====*/

```

```

Dandelin := proc(var_a, var_b, var_c, var_d, var_alpha)
local a, b, c, d, alpha, N, A, eps, omega, lista_immagini, viewBox,
zS1, rS1, zS2, rS2, zC1, rC1, zC2, rC2, h, h1, h2, r, r1, r2, t1, t2,
conol, cono2, sferal, sfera2, circonferenza, circonferenzal,
fuocol, fuoco2, piano, pianol, piano2, direttriciel, direttrice2;

begin

// Conversione dei dati di input in numeri in virgola mobile
a := float(var_a):
b := float(var_b):
c := float(var_c):
d := float(var_d):
alpha := float(var_alpha*PI/180): // l'angolo di apertura in radianti

// Controllo dell'esistenza del piano.
if ([a, b, c] = [0.0, 0.0, 0.0]) then
error("Non è stato definito alcun piano!");
end_if;

// Controllo dell'ampiezza dell'angolo di apertura
if ((alpha <= 0.0) or (alpha >= float(PI/2))) then
error("L'angolo di apertura del cono non è valido!");
end_if;

```

```

// Alcune variabili ausiliarie
N := float(a^2 + b^2 + c^2):
A := float(N*sin(alpha)^2 - c^2):
eps := 1.0e-6: // variabile per approssimare lo 0

// L'angolo tra il piano e l'asse del cono
omega := abs(float(PI/2 - arccos(c/sqrt(N)))):

// Il piano passante per il cono
if (a <> 0.0) then
  piano := plot::Plane([-d/a, 0, 0], [a, b, c], Color = RGB::Yellow, Mesh = [4, 4]):
elif (b <> 0.0) then
  piano := plot::Plane([0, -d/b, 0], [a, b, c], Color = RGB::Yellow, Mesh = [4, 4]):
else
  piano := plot::Plane([0, 0, -d/c], [a, b, c], Color = RGB::Yellow, Mesh = [4, 4]):
end_if;

// Lista che contiene le immagini da visualizzare
lista_immagini := [piano]:

/*****\
| Coniche degeneri |
\*****/
if (d = 0.0) then
  h := 10.0:
  r := h*tan(alpha):
  cono1 := plot::Cone(r, [0, 0, h], [0, 0, 0], Color=RGB::Blue.[0.3]):
  cono2 := plot::Cone(r, [0, 0, -h], [0, 0, 0], Color=RGB::Blue.[0.3]):
  viewBox := [-r..r, -r..r, -h..h]:
  lista_immagini := lista_immagini.[cono1, cono2]:

/*****\
| Coniche non degeneri |
\*****/
else
  /*****\
  | Parabola non degenera |
  \*****/
  if (abs(omega - alpha) < eps) then

    // La sfera di Dandelin
    zS1 := -d/(2*c): // quota del centro della sfera di Dandelin
    rS1 := abs(zS1*sin(alpha)): // raggio della sfera di Dandelin
    sferal := plot::Sphere(rS1, [0,0, zS1], Color = RGB::Green.[0.5]):

    // Il cono che contiene la sfera di Dandelin
    h := (abs(zS1) + rS1)*1.25: // altezza del cono
    r := h*tan(alpha): // raggio del cono
    if (zS1 > 0.0) then
      cono1 := plot::Cone(r, [0, 0, h], [0, 0, 0], Color=RGB::Blue.[0.3]):

```

```

    viewBox := [-r..r, -r..r, 0..h]:
else
    cono1 := plot::Cone(r, [0, 0, -h], [0, 0, 0], Color=RGB::Blue.[0.3]):
    viewBox := [-r..r, -r..r, -h..0]:
end_if;

// La circonferenza di tangenza tra il cono e la sfera di Dandelin
zC1 := zS1/(1 + tan(alpha)^2): // quota del centro della circonferenza di tangenza
rC1 := float(rS1*cos(alpha)): // raggio della circonferenza di tangenza
circonferenzal := plot::Curve3d([rC1*cos(t), rC1*sin(t), zC1], t=0..2*PI, Color = RGB::Black, LineWidth=0.75):

// Il piano passante per la circonferenza di tangenza tra il cono e la sfera di Dandelin
piano1 := plot::Plane([0, 0, zC1], [0, 0, 1], Color = RGB::Orange.[0.4], Mesh = [4, 4]):

// La direttrice
if (a <> 0.0) then
    direttriciel := plot::Curve3d([(-b*t -c*zC1 - d)/a, t, zC1], t=-100..100):
elif (b <> 0.0) then
    direttriciel := plot::Curve3d([t, (-a*t -c*zC1 - d)/b, zC1], t=-100..100):
end_if;

// Il fuoco
solve(a*a*t + b*b*t + c*(zS1 + c*t) + d = 0.0, t):
t1 := %[1]:
fuoco1 := plot::Point3d([a*t1, b*t1, zS1 + c*t1]):

// Tutti gli oggetti da visualizzare
lista_immagini := lista_immagini.[cono1, sferal, piano1, circonferenzal, direttriciel, fuoco1]:

/*****\
| Circonferenza non degenerare |
\*****/
elif ((a = 0.0) and (b = 0.0)) then

    // La sfera di Dandelin
    zS1 := (c*d + sqrt(d^2*N*sin(alpha)^2))/A: // quota del centro della sfera di Dandelin
    rS1 := abs(zS1*sin(alpha)): // raggio della sfera di Dandelin
    sferal := plot::Sphere(rS1, [0,0, zS1], Color = RGB::Green.[0.5]):

    // Il cono che contiene la sfera di Dandelin
    h := (abs(zS1) + rS1)*1.25: // altezza del cono
    r := h*tan(alpha): // raggio del cono
    if (zS1 > 0.0) then
        cono1 := plot::Cone(r, [0, 0, h], [0, 0, 0], Color=RGB::Blue.[0.3]):
        viewBox := [-r..r, -r..r, 0..h]:
    else
        cono1 := plot::Cone(r, [0, 0, -h], [0, 0, 0], Color=RGB::Blue.[0.3]):
        viewBox := [-r..r, -r..r, -h..0]:
    end_if;

```

```

// La circonferenza di tangenza tra il cono e la sfera di Dandelin
zC1 := zS1/(1 + tan(alpha)^2): // quota del centro della circonferenza di tangenza
rC1 := float(rS1*cos(alpha)): // raggio della circonferenza di tangenza
circonferenzal := plot::Curve3d([rC1*cos(t), rC1*sin(t), zC1], t=0..2*PI, Color = RGB::Black, LineWidth=0.75):

// Il (doppio) "fuoco"
solve(a*a*t + b*b*t + c*(zS1 + c*t) + d = 0.0, t):
t1 := %[1]:
fuocol := plot::Point3d([0, 0, -d/c]):

// Il piano passante per la circonferenza di tangenza tra il cono e la prima sfera di Dandelin
pianol := plot::Plane([0, 0, zC1], [0, 0, 1], Color = RGB::Orange.[0.4], Mesh = [4, 4]):

// Tutti gli oggetti da visualizzare
lista_immagini := lista_immagini.[conol, sferal, circonferenzal, pianol, fuocol]:

/*****\
| Ellisse o iperbole non degenerare |
\*****/
else

// La prima sfera di Dandelin
zS1 := (c*d + sqrt(d^2*N*sin(alpha)^2))/A: // quota del centro della prima sfera di Dandelin
rS1 := abs(zS1*sin(alpha)): // raggio della prima sfera di Dandelin
sferal := plot::Sphere(rS1, [0,0, zS1], Color = RGB::Green.[0.5]):

// La seconda sfera di Dandelin
zS2 := (c*d - sqrt(d^2*N*sin(alpha)^2))/A: // quota del centro della seconda sfera di Dandelin
rS2 := abs(zS2*sin(alpha)): // raggio della seconda sfera di Dandelin
sfera2 := plot::Sphere(rS2, [0,0, zS2], Color = RGB::Green.[0.5]):

if (alpha < omega) then
// Ellisse: il cono che contiene le sfere di Dandelin
zMax := max(abs(zS1), abs(zS2)):
rMax := max(rS1, rS2):
h := (zMax + rMax)*1.25: // altezza del cono
r := h*tan(alpha): // raggio del cono
if (zS1 > 0.0) then
conol := plot::Cone(r, [0, 0, h], [0, 0, 0], Color=RGB::Blue.[0.3]):
viewBox := [-r..r, -r..r, 0..h]:
else
conol := plot::Cone(r, [0, 0, -h], [0, 0, 0], Color=RGB::Blue.[0.3]):
viewBox := [-r..r, -r..r, -h..0]:
end_if;

// Tutti gli oggetti da visualizzare
lista_immagini := lista_immagini.[conol]:
else
// Iperbole: il (doppio) cono che contiene le sfere di Dandelin

```

```

h1 := (abs(zS1) + rS1)*1.25: // altezza del primo cono
r1 := h1*tan(alpha): // raggio del primo cono
h2 := (abs(zS2) + rS2)*1.25: // altezza del secondo cono
r2 := h2*tan(alpha): // raggio del secondo cono
r := max(r1, r2):
if (zS1 > 0.0) then
  cono1 := plot::Cone(r1, [0, 0, h1], [0, 0, 0], Color=RGB::Blue.[0.3]):
  cono2 := plot::Cone(r2, [0, 0, -h2], [0, 0, 0], Color=RGB::Blue.[0.3]):
  viewBox := [-r..r, -r..r, -h2..h1]:
else
  cono1 := plot::Cone(r1, [0, 0, -h1], [0, 0, 0], Color=RGB::Blue.[0.3]):
  cono2 := plot::Cone(r2, [0, 0, h2], [0, 0, 0], Color=RGB::Blue.[0.3]):
  viewBox := [-r..r, -r..r, -h1..h2]:
end_if;

// Tutti gli oggetti da visualizzare
lista_immagini := lista_immagini.[cono1, cono2]:
end_if;

// La circonferenza di tangenza tra il cono e la prima sfera di Dandelin
zC1 := zS1/(1 + tan(alpha)^2): // quota del centro della circonferenza di tangenza
rC1 := float(rS1*cos(alpha)): // raggio della circonferenza di tangenza
circonferenza1 := plot::Curve3d([rC1*cos(t), rC1*sin(t), zC1], t=0..2*PI, Color = RGB::Black, LineWidth=0.75):

// La circonferenza di tangenza tra il cono e la seconda sfera di Dandelin
zC2 := zS2/(1 + tan(alpha)^2): // quota del centro della circonferenza di tangenza
rC2 := float(rS2*cos(alpha)): // raggio della circonferenza di tangenza
circonferenza2 := plot::Curve3d([rC2*cos(t), rC2*sin(t), zC2], t=0..2*PI, Color = RGB::Black, LineWidth=0.75):

// Il piano passante per la circonferenza di tangenza tra il cono e la prima sfera di Dandelin
piano1 := plot::Plane([0, 0, zC1], [0, 0, 1], Color = RGB::Orange.[0.4], Mesh = [4, 4]):

// Il piano passante per la circonferenza di tangenza tra il cono e la seconda sfera di Dandelin
piano2 := plot::Plane([0, 0, zC2], [0, 0, 1], Color = RGB::Orange.[0.4], Mesh = [4, 4]):

// La prima direttrice
if (a <> 0.0) then
  direttrice1 := plot::Curve3d([(-b*t -c*zC1 - d)/a, t, zC1], t=-100..100):
elif (b <> 0.0) then
  direttrice1 := plot::Curve3d([t, (-a*t -c*zC1 - d)/b, zC1], t=-100..100):
end_if;

// La seconda direttrice
if (a <> 0.0) then
  direttrice2 := plot::Curve3d([(-b*t -c*zC2 - d)/a, t, zC2], t=-100..100):
elif (b <> 0.0) then
  direttrice2 := plot::Curve3d([t, (-a*t -c*zC2 - d)/b, zC2], t=-100..100):
end_if;

// Il primo fuoco

```

```
solve(a*a*t + b*b*t + c*(zS1 + c*t) + d = 0.0, t):
t1 := %[1]:
fuoco1 := plot::Point3d([a*t1, b*t1, zS1 + c*t1]):

// Il secondo fuoco
solve(a*a*t + b*b*t + c*(zS2 + c*t) + d = 0.0, t):
t2 := %[1]:
fuoco2 := plot::Point3d([a*t2, b*t2, zS2 + c*t2]):

// Tutti gli oggetti da visualizzare
lista_immagini := lista_immagini.[sfera1, sfera2, piano1, piano2, circonferenza1,
                                   circonferenza2, direttrice1, direttrice2, fuoco1, fuoco2]:
end_if;
end_if;

// Visualizzazione degli oggetti
plot(op(lista_immagini), ViewingBox = viewBox, Axes = None);
end_proc;
```