

Projekt Dreiecksmitte

Michael Nüsken, 3. Dezember 2002

```
> restart:
with(LinearAlgebra):
with(plots):
with(plottools):
Warning, the name changecoords has been redefined
Warning, the name arrow has been redefined
```

Wir speichern die Ecken des Dreiecks als Vektoren.

```
> A:=<0,0>:
B:=<sqrt(3)/2,3/2>:
C:=<-sqrt(3)/2,3/2>:
```

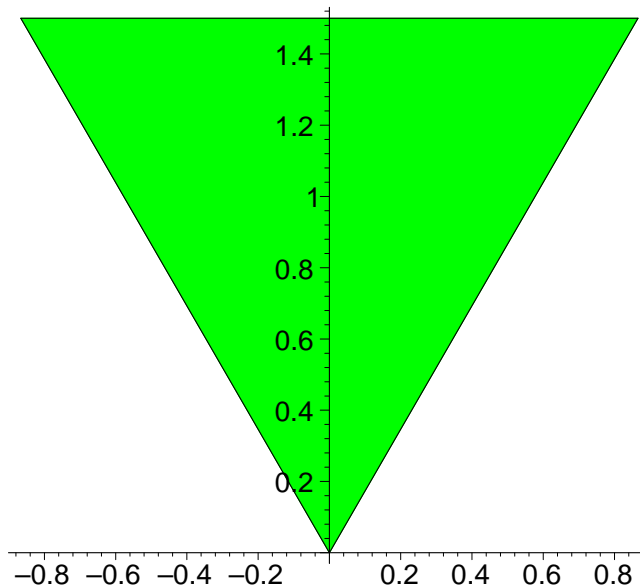
Und weil's so schön ist berechnen wir gleich den Schwerpunkt.

```
> S:=(A+B+C)/3;
```

$$S := \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Und wie sieht das nun aus? Zeichnen wir:

```
> plotDreieck:=polygonplot(
  map(convert,[A,B,C],list),
  color=green, scaling=constrained):
plotDreieck;
```



```
>
```

Jetzt müssen wir ein wenig **nachdenken** ...

- Wir sollen die Fläche mit Geraden halbieren. (Wie kriegt man bloß die Fläche eines schrägen Dreiecks raus?)
- Dazu müssen wir wohl eine allgemeine Gerade mit zwei der Seiten schneiden. Geraden können v

gut durch Geradengleichungen $y = ax + b$ darstellen (gespeichert als Gleichung mit Unbestimmten x und y).

- Um die Geraden zu den Seiten zu bekommen, müssen wir Punkte (gespeichert als Vektoren) verbinden.

Ok, da läßt sich anfangen:

Diese Prozedur bestimmt die Gleichung der Geraden durch Punkte P und Q. (Außer wenn...)

```
> verbindungsgerade:=proc(P,Q)
  global x,y;
  (y-P[2])/(x-P[1])=(Q[2]-P[2])/(Q[1]-P[1]);
  isolate(%,y);
end proc;
```

Als Beispiel bestimmen wir die beiden Seitenhalbierenden durch B und C.

```
> verbindungsgerade(S,B),verbindungsgerade(S,C);
```

$$y = \frac{\sqrt{3}x}{3} + 1, y = -\frac{\sqrt{3}x}{3} + 1$$

Diese Prozedur bestimmt den Schnittpunkt zweier Geraden.

```
> Schnittpunkt:=proc(g,h)
  global x,y;
  solve({g,h},{x,y});
  subs(%,<x,y>);
end proc;
```

Das ist die allgemeine Gerade, mit der wir gleich das Dreieck in zwei Teile zerlegen wollen.

```
> g:= y=a*x+b;
```

$$g := y = ax + b$$

Wir bestimmen die Schnittpunkte P bzw. Q der allgemeinen Geraden mit AB bzw. AC. Das Dreieck APQ ist nun das von der allgemeinen Gerade von unserem Dreieck ABC abgeschnittene Dreieck, jedenfalls dann, wenn die y-Koordinaten von P und Q beide zwischen 0 und 3/2 liegen, also auf der Strecke AB bzw. AC. Im Halbierungsfalle können wir das jedoch leicht genau sagen ..

```
> P:=Schnittpunkt( g, verbindungsgerade(A,B) );
  Q:=Schnittpunkt( g, verbindungsgerade(A,C) );
  P,Q;
```

$$\left[\begin{array}{c} -\frac{b}{a-\sqrt{3}} \\ \frac{\sqrt{3}b}{a-\sqrt{3}} \end{array} \right] \left[\begin{array}{c} -\frac{b}{a+\sqrt{3}} \\ \frac{\sqrt{3}b}{a+\sqrt{3}} \end{array} \right]$$

Bestimme allgemein die Fläche des Dreiecks APQ.

```
> fläche:=unapply( 1/2*Determinant(<P|Q>), a,b );
```

Hoppla, das ist aber einfach.

$$fläche = (a, b) \rightarrow -\frac{b^2 \sqrt{3}}{(a-\sqrt{3})(a+\sqrt{3})}$$

Für $a = 0$, $b = \frac{3}{2}$ erhalten wir die Gerade BC,

```
> simplify(verbindungsgerade(B,C));
```

$$y = \frac{3}{2}$$

also wird die obige Prozedur damit die Fläche des ganzen Dreiecks angeben. Damit können wir nu die Bedingung aufstellen dafür, dass das Dreieck APQ gerade halb so gross ist wie das ursprünglich

Wir verwenden das Ergebnis, um die allgemeine Gerade so zu spezialisieren, dass $h(a)$

Halbierungsgeraden sind. Wir bemerken, dass nur die Steigungen erlaubt sind, die zwischen den

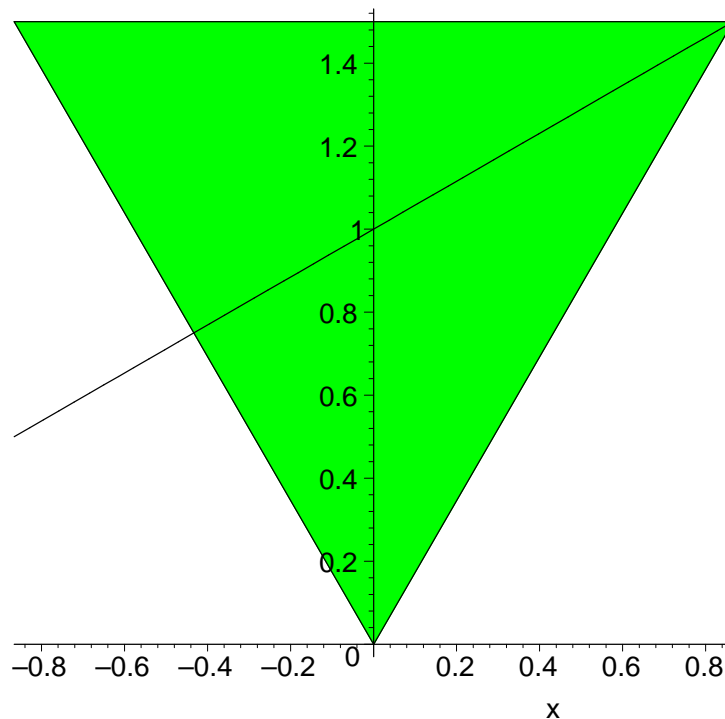
beiden oben berechneten Winkelhalbierenden liegen, also $a = -\frac{1}{\sqrt{3}} \dots \frac{1}{\sqrt{3}}$.

```
> fläche(a,b) = 1/2 * fläche(0,3/2):  
isolate(% ,b):  
convert(% ,radical):  
h:=unapply( subs(% ,g), a );
```

$$h := a \rightarrow y = ax + \sqrt{-\frac{3}{8}a^2 + \frac{9}{8}}$$

Schön. Zeichnen wir mal ein wenig:

```
> plotHalbierungsgerade:=  
  a -> plot( subs(h(a),y), x=C[1]..B[1], color=black );  
> display( [ plotDreieck, plotHalbierungsgerade(1/sqrt(3)) ] );
```

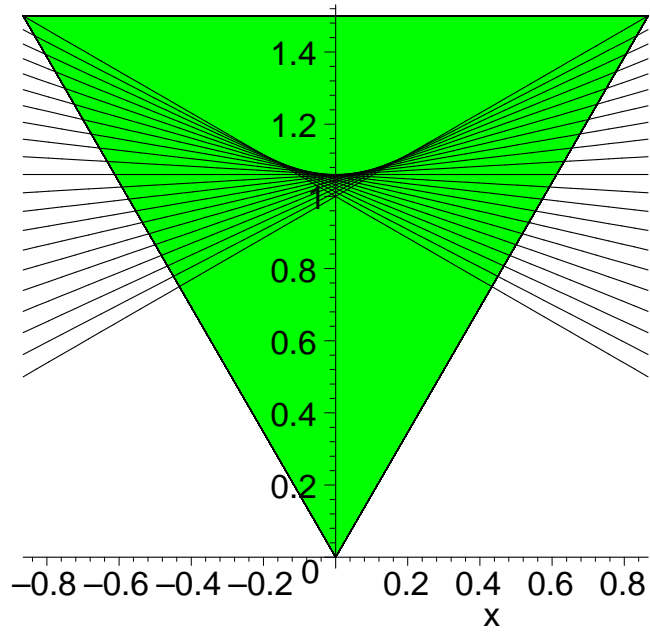


Vielleicht mal ein paar auf ein Mal?

```
> L:=NULL:  
for i from 0 to 1 by 1/20 do  
  L:=L,display( [  
    plotDreieck,  
    plotHalbierungsgerade( (2*i-1)/sqrt(3) )  
  ],  
  scaling=constrained );  
end do:
```

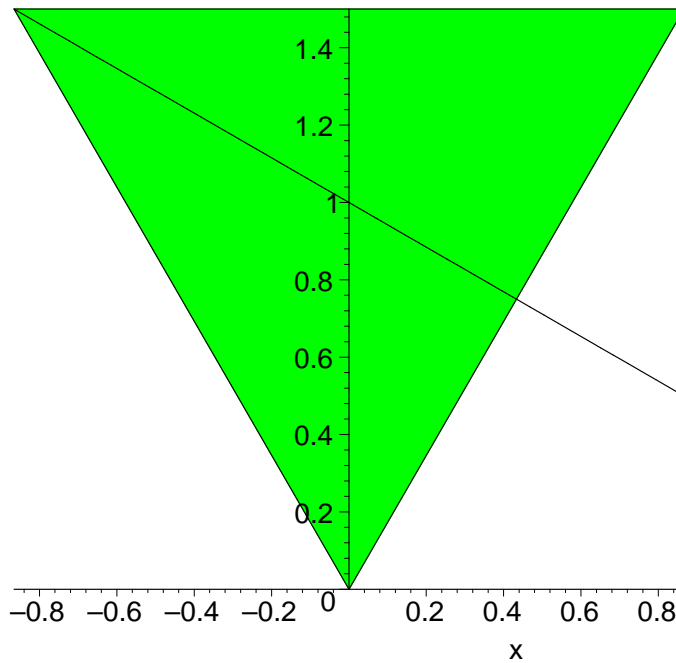
```
display( [L] );
```

Aha:



Wie wär's mit einem Film??

```
> display( [L], insequence=true, scaling=constrained );
```



Die "Hüllkurve" bekommen wir so: wir schneiden $h(a)$ mit einer anderen Geraden $h(a + \delta)$ und lassen dann δ gegen 0 laufen. Der so gefundene Punkt gehört zur Hüllkurve.

```
> Schnittpunkt( h(a), h(a+delta) ):
%[1]:
x=limit( %, delta=0 );
parametrisierteKurve:=simplify([ %, subs( %, h(a) ) ]);
```

$$x = \frac{3a}{2\sqrt{-6a^2 + 18}}$$

$$parametrisierteKurve := \left[x = \frac{3a}{2\sqrt{-6a^2 + 18}}, y = \frac{9}{2\sqrt{-6a^2 + 18}} \right]$$

Jetzt formen wir die mit a parametrisierten Hüllkurvenpunkte in eine Funktionsgleichung um.

```
> isolate( parametrisierteKurve[1], a ):
subs( %, parametrisierteKurve[2] ):
Kurve:=simplify(%) assuming real;
```

$$Kurve := y = \frac{\sqrt{6}\sqrt{8x^2 + 3}}{4}$$

Ein anderes Verständnis ...

```
> map( z->z^2, Kurve ):
isolate(%, 9/8);
```

$$\frac{9}{8} = y^2 - 3x^2$$

Wir rechnen das erlaubte Intervall für y in ein Intervall für x um, nachdem wir uns überzeugt haben, dass x streng monoton von a abhängt.

```

> simplify( diff( rhs(parametrisierteKurve[1]), a ) ):
% * (3-a^2)^(3/2):
simplify(%) assuming real:
% / (3-a^2)^(3/2):
%>0, a=-1/sqrt(3)..1/sqrt(3);

subs(a=-1/sqrt(3),parametrisierteKurve[1]):
subs(a=+1/sqrt(3),parametrisierteKurve[1]):
xKurve:=simplify(subs(%,x)..subs(%,x));

```

$$0 < \frac{3\sqrt{6}}{4(-a^2+3)^{(3/2)}, a = -\frac{\sqrt{3}}{3} \dots \frac{\sqrt{3}}{3}}$$

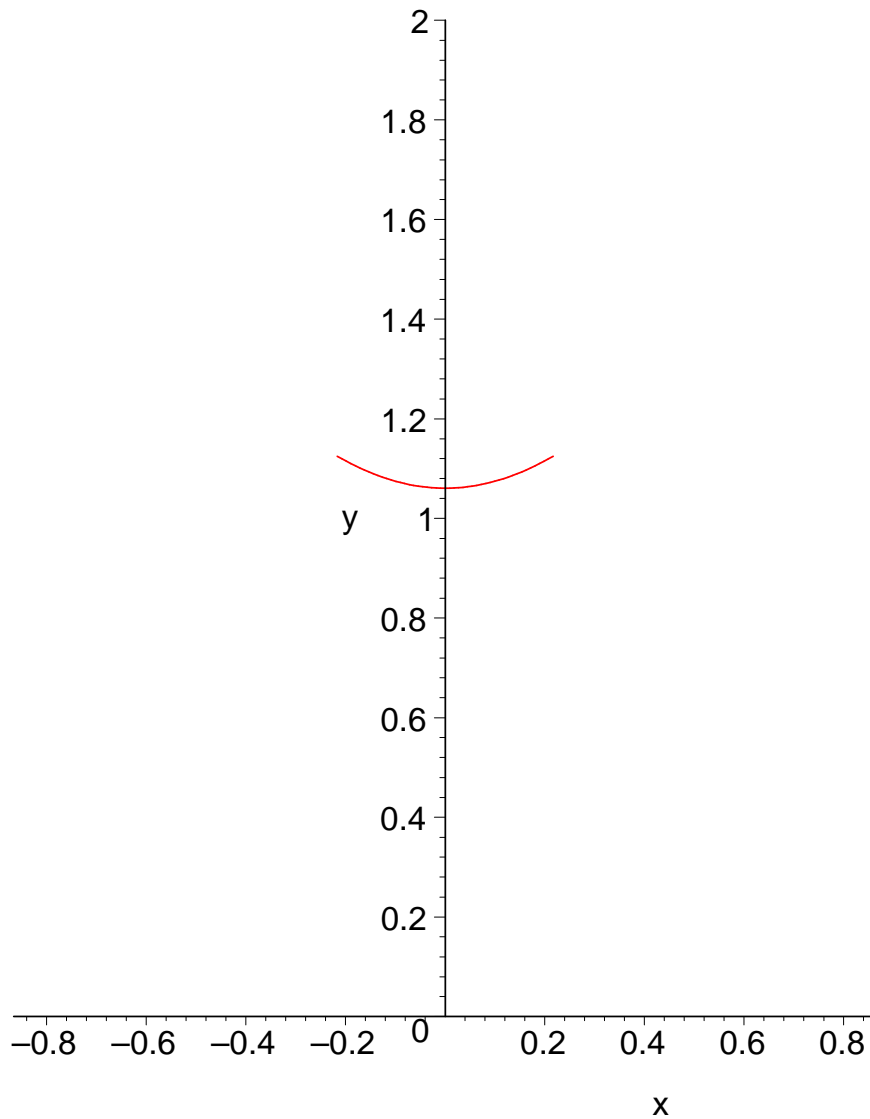
$$xKurve := -\frac{\sqrt{3}}{8} \dots \frac{\sqrt{3}}{8}$$

Und jetzt zeichnen wir das Ergebnis. Zuerst zeichnen wir die berechnete Hüllkurve und speichern das.

```

> plotKurve:=plot(
  rhs(Kurve),
  x=xKurve,
  y=0..2,
  view=[-sqrt(3)/2..sqrt(3)/2,0..2],
  scaling=constrained ):
% ;

```

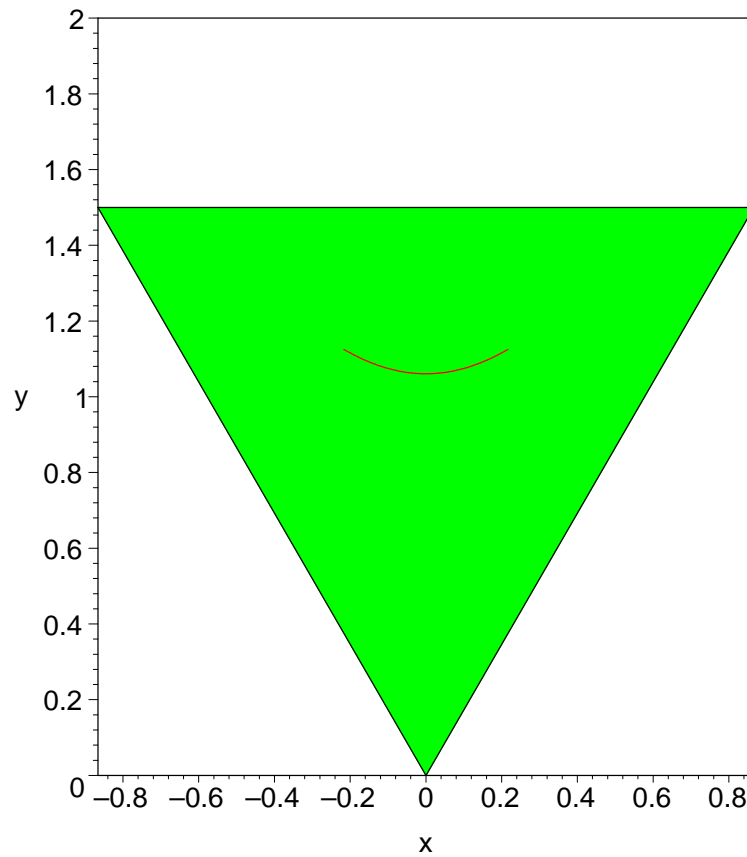


Wir merken uns mal die Plotoptionen, die wir ständig brauchen: Das richtige Fenster und die Forderung nach gleichen Einheiten.

```
> plotOptionen:=
  view=[-sqrt(3)/2..sqrt(3)/2,0..2],
  scaling=constrained,
  axes=boxed:
```

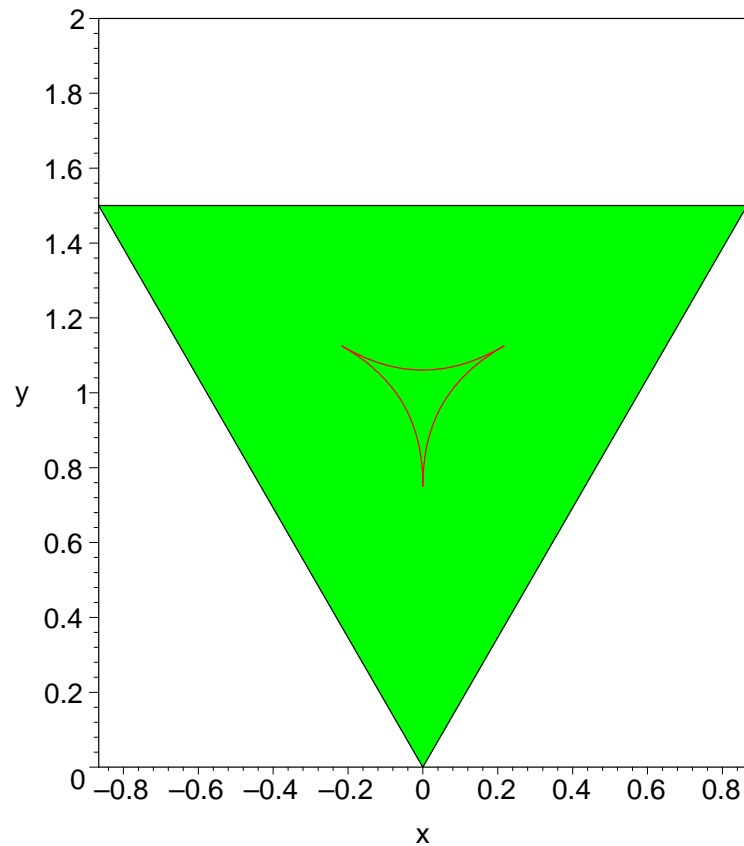
Schauen wir uns damit nochmal die Kurve an, diesmal gleich mit dem Dreieck.

```
> display( [ plotDreieck, plotKurve ], plotOptionen );
```



So und jetzt noch einmal mit den gedrehten Versionen, damit wir ein vollständiges Bild bekommen

```
> plotBasis:=display([
  plotDreieck,
  plotKurve,
  rotate( plotKurve, 2*Pi/3, convert(S,list) ),
  rotate( plotKurve, 4*Pi/3, convert(S,list) )
], plotOptionen):
%i
```



Wir zeichnen einige der gefundenen Halbierungsgerade und speichern das.

```
> plotHalbierungsgeradenschar :=
  display(
    [seq(
      plotHalbierungsgerade(i/10/sqrt(3)),
      i=-10..10
    )]
  ):

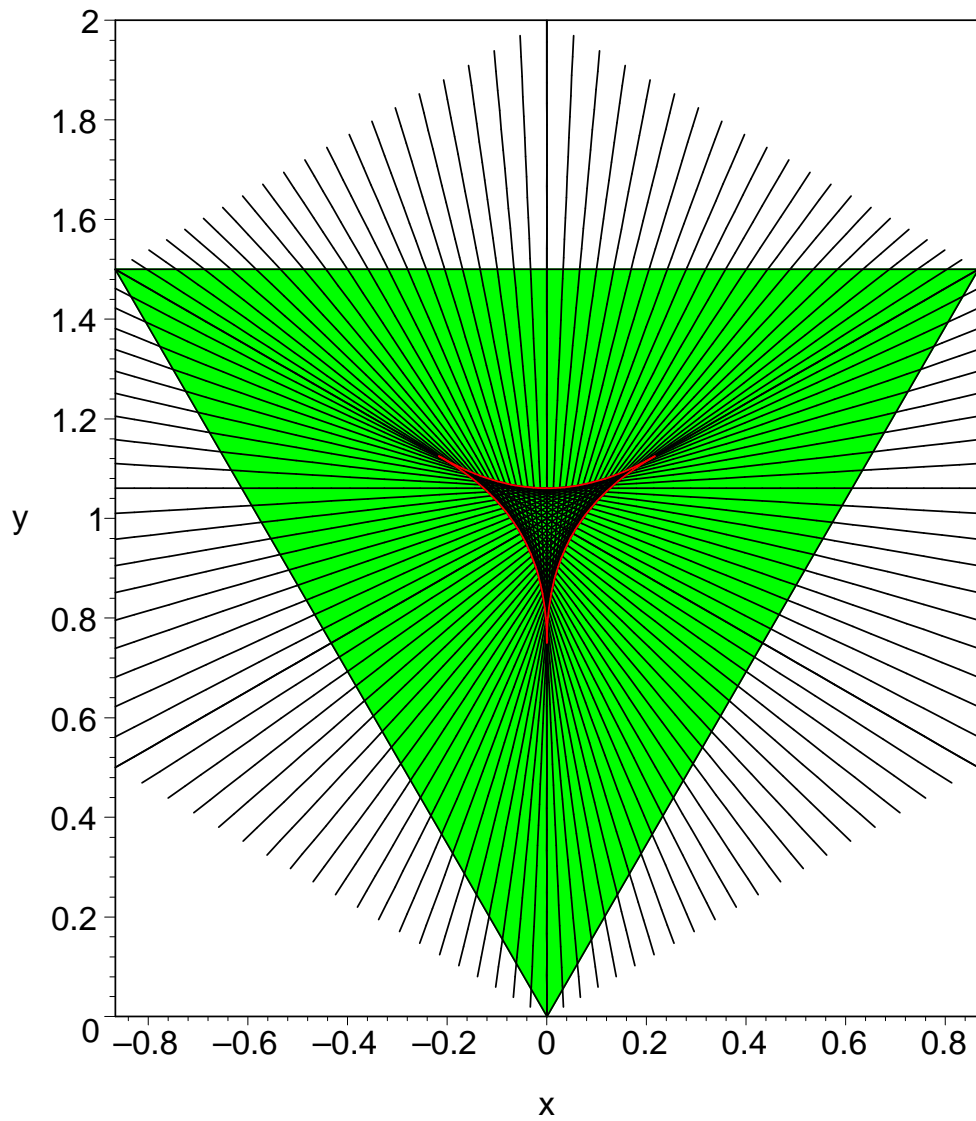
```

Nun erzeugen wir ein Bild mit dem Dreieck und je drei Kopien der Hüllkurve und der

Geradenschar, die um die Winkel 0 , $\frac{2\pi}{3}$ und $\frac{4\pi}{3}$ gedreht sind.

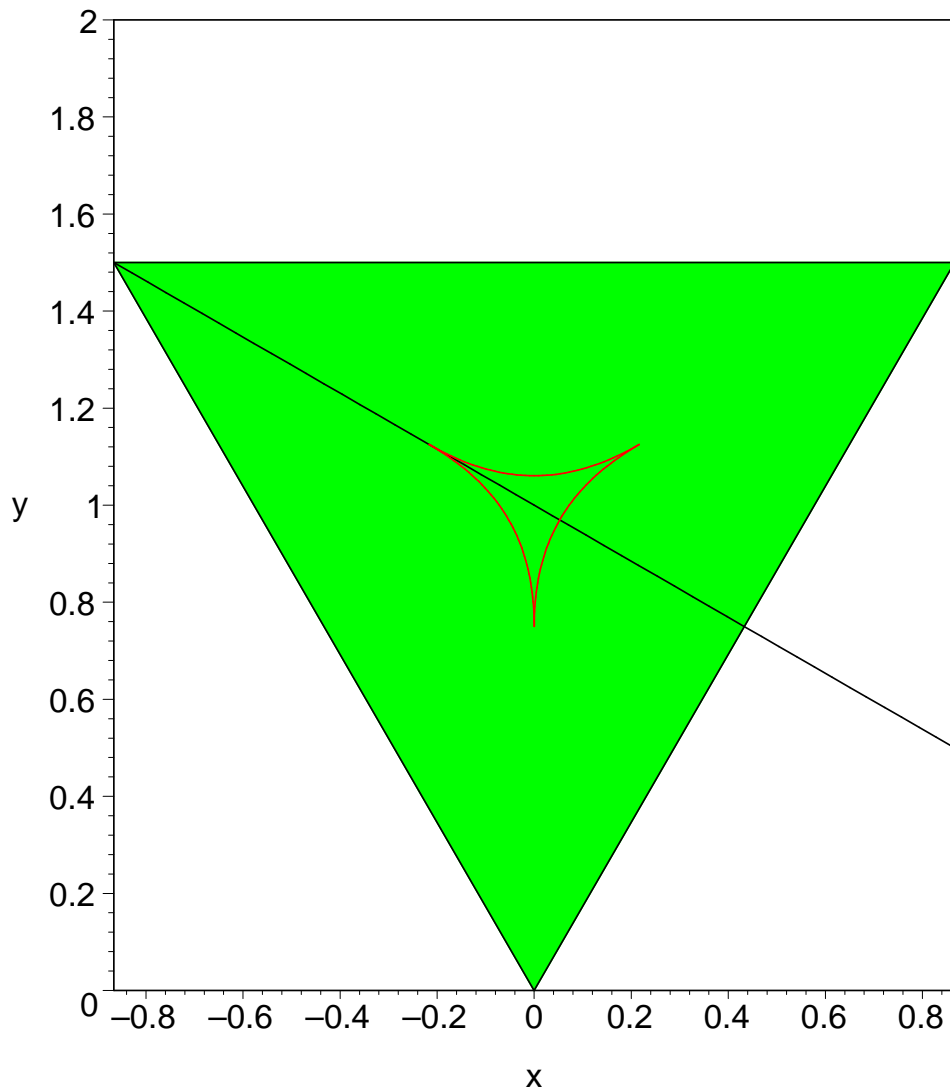
```
> display([
  plotBasis,
  plotHalbierungsgeradenschar,
  rotate( plotHalbierungsgeradenschar, 2*Pi/3, convert(S,list)
),
  rotate( plotHalbierungsgeradenschar, 4*Pi/3, convert(S,list)
),
  NULL], plotOptionen);

```



Zum Abschluß nochmal als Film:

```
> L:=NULL:
  for r from 0 to 2 do
    for i from 0 to .999999999 by 1/50 do
      L:=L, display([
        plotBasis,
        rotate( plotHalbierungsgerade( (2*i-1)/sqrt(3) ),
        -r*2*Pi/3, convert(S,list) )
      ], plotOptionen);
    end do:
  end do:
> display([L],insequence=true,scaling=constrained);
```



[>

– Noch ein kleiner Nachschlag

Die Kurve am Rande ...

```
> simplify(subs(x=lhs(xKurve),Kurve)),
  simplify(subs(x=rhs(xKurve),Kurve));
```

Eine Fläche ... Bloß welche? Kriegt Ihr heraus, was ich hier gemacht habe?

[Und was bewirkt 'Int' im Unterschied zu 'int'? Wozu 'value'?]

```
> 3*(
  1/2*(rhs(xKurve)-lhs(xKurve))*(9/8-1)
  -
  Int(simplify(9/8-rhs(Kurve)),x=xKurve)
);
```

```
F:=simplify(value(%));
```

??

```
> [F,fläche(0,3/2),fläche(0,3/2)/F];
  evalf(%);
```

[>