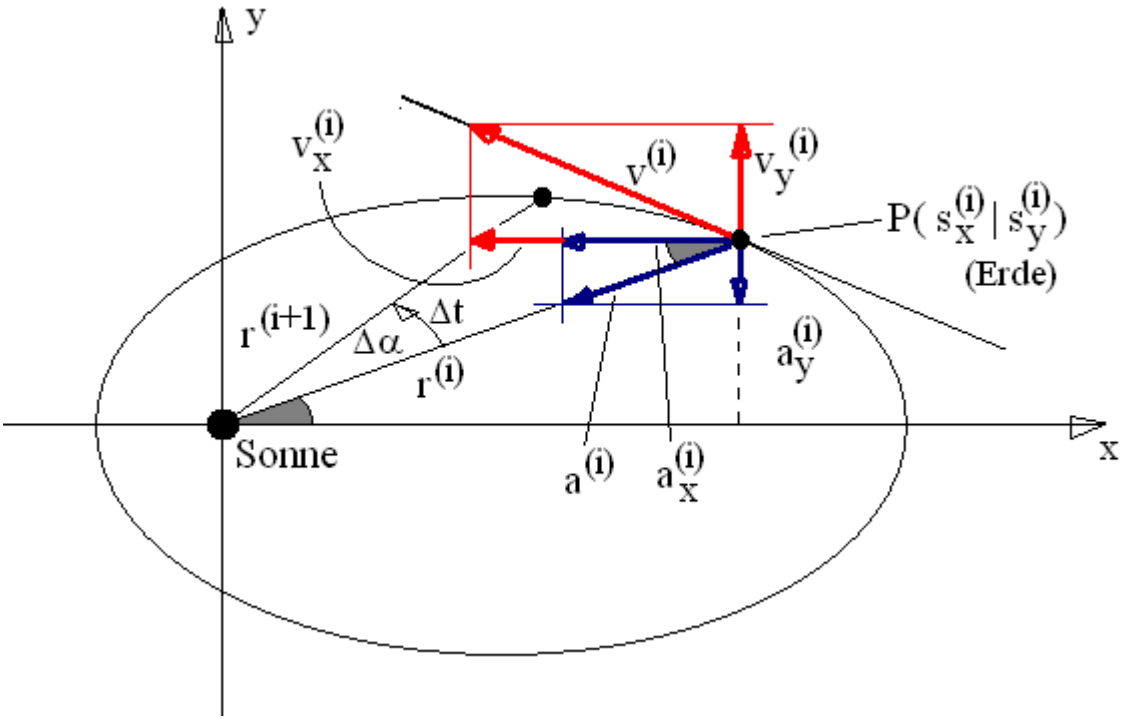
Simulation einer Planetenbewegung - Orientierung am Beispiel Sonne-Erde

Die Erde bewegt sich auf einer Ellipsenbahn um die Sonne, die sich in einem der Brennpunkte befindet. In der Skizze ist die Ellipse übertrieben gezeichnet. In Wahrheit könnte man im Maßstab der Skizze die tatsächliche Erdbahn-Ellipse von einem idealen Kreis nicht unterscheiden – so gering ist die s.g. Exzentrizität der Ellipse. Es gehört zu den großen Leistungen der Astronomie, dass Kepler die Ellipsenform rechnerisch aus den Daten von Tycho Brahe bestimmen konnte.

Ausgangslage:
Die Erde befindet sich im Punkt $P^{(i)} = P(s_x^{(i)} | s_y^{(i)})$ und bewegt sich im folgenden Zeitintervall Δt um den Winkel $\Delta\alpha$ zum Punkt $P^{(i+1)} = P(s_x^{(i+1)} | s_y^{(i+1)})$.
Im Punkt $P^{(i)}$ wirkt auf die Erde die Zentripetalbeschleunigung $a^{(i)}$ in Richtung Sonne.
(Bemerkung: In der Skizze werden alle Vektorpfeile der Lesbarkeit halber weggelassen.)
Auf dem Weg der Erde entlang der Ellipse zum Punkt $P^{(i+1)}$ ändert sich diese Beschleunigung kontinuierlich.
Für unsere Simulation machen wir jetzt *bewusst den Fehler* anzunehmen, dass sich die Beschleunigung im Zeitintervall Δt *nicht ändert*, weder nach Betrag noch nach Richtung. Dann nämlich sind wir in der Lage, nach Aufspaltung der Beschleunigung in ihre x- und y-Komponente mit den Gesetzen für die geradlinig-beschleunigte Bewegung die nächste Position $P^{(i+1)}$ berechnen zu können.



Gleichungen für die x-Richtung:
In Punkt $P^{(i)}$ besitzt der Planet die Geschwindigkeit $v_x^{(i)}$, die sich im Laufe des Intervalls Δt um $a_x^{(i)} \cdot \Delta t$ auf $v_x^{(i+1)}$ erhöht, also gilt $v_x^{(i+1)} = v_x^{(i)} + a_x^{(i)} \cdot \Delta t$. Nach den Gesetzten der Dynamik ändert sich der Ort $s_x^{(i)}$ aufgrund der Anfangsgeschwindigkeit um $v_x^{(i)} \cdot \Delta t$ und aufgrund der Beschleunigung um $\frac{1}{2} \cdot a_x^{(i)} \cdot (\Delta t)^2$. Also gilt insgesamt

$s_x^{(i+1)} = s_x^{(i)} + v_x^{(i)} \cdot \Delta t + \frac{1}{2} \cdot a_x^{(i)} \cdot (\Delta t)^2$. Mit derselben Idee erhält man in y-Richtung $s_y^{(i+1)} = s_y^{(i)} + v_y^{(i)} \cdot \Delta t + \frac{1}{2} \cdot a_y^{(i)} \cdot (\Delta t)^2$.

Wie man sieht, wird die neue Geschwindigkeit in beide Richtungen Schritt für Schritt aus der jeweils vorherigen berechnet. Die Gesamtgeschwindigkeit (in Richtung der jeweiligen Tangente) wird nicht benötigt.

Anders verhält es sich mit der Beschleunigung: An jedem neuen Ort muss zunächst die Beschleunigung zur Sonne berechnet werden, danach wird sie in die Komponenten aufgespalten. Das soll im Folgenden geschehen:

Die beiden in der Skizze eingefärbten Winkel sind gleich groß, sie werden mit β bezeichnet.

Dann gilt einerseits $a_x^{(i)} = a^{(i)} \cdot \cos(\beta)$ und andererseits $\cos(\beta) = \frac{s_x^{(i)}}{r^{(i)}}$, folglich

$a_x^{(i)} = a^{(i)} \cdot \frac{s_x^{(i)}}{r^{(i)}}$. Entsprechend gilt für die y-Komponnte $a_y^{(i)} = a^{(i)} \cdot \frac{s_y^{(i)}}{r^{(i)}}$.

Nun können wir alle Berechnungen für 1 Schleifendurchgang der Simulation, der dann ein paar Tausend mal wiederholt wird, zusammenstellen:

1. Neuen Radius ausrechnen: $r^{(i)} = \sqrt{(s_x^{(i)})^2 + (s_y^{(i)})^2}$

2. Neue Beschleunigung ausrechnen: $a^{(i)} = \frac{C}{(r^{(i)})^2}$, C ist definiert als $\gamma \cdot m_S$

   (folgt aus Gravitationsges. $F_Z = \gamma \cdot \frac{m_P \cdot m_S}{r^2}$)

3. Neue Komponenten der Beschl.: $a_x^{(i)} = a^{(i)} \cdot \frac{s_x^{(i)}}{r^{(i)}}$, $a_y^{(i)} = a^{(i)} \cdot \frac{s_y^{(i)}}{r^{(i)}}$

4. Neue Komponenten des Ortes: $s_x^{(i+1)} = s_x^{(i)} + v_x^{(i)} \cdot \Delta t + \frac{1}{2} \cdot a_x^{(i)} \cdot (\Delta t)^2$

   $s_y^{(i+1)} = s_y^{(i)} + v_y^{(i)} \cdot \Delta t + \frac{1}{2} \cdot a_y^{(i)} \cdot (\Delta t)^2$

5. Neue Komponenten der Geschw.: $v_x^{(i+1)} = v_x^{(i)} + a_x^{(i)} \cdot \Delta t$, $v_y^{(i+1)} = v_y^{(i)} + a_y^{(i)} \cdot \Delta t$

Jetzt geht's wieder zu Punkt 1, wobei nun die neuen Ortskomponenten eingesetzt werden. Die Ortskomponenten werden in einer Liste gespeichert, damit man sie alle zeichnen kann. Die Geschwindigkeits- und Beschleunigungswerte müssen nicht gespeichert werden und können bei der Berechnung überschrieben werden. Es folgt der gesamte Quelltext der Programmierung mit MuPAD.

Quelltext:

- ```
  reset():;DIGITS:=6:;
  ```

- ```
  mS:=1.989*10^30:;  //Masse der Sonne in kg
  f:=6.67259*10^-11:; // Gravitationskonstante
  C:=float(f*mS):;
  ```

- ```
  Bahn:=proc(sx0,sy0,vx0,vy0,n,dt)
  local vx,vy,a,ax,ay,d,i,ok;
  begin
  print("Geschwindigkeit im Aphel:",vy0,"m/s");
  sx[0]:=sx0;sy[0]:=sy0;
  vx:=vx0;vy:=vy0;
  s:=0;ok:=TRUE;
  for i from 0 to n do
    d:=float(sqrt(sx[i]^2+sy[i]^2));
    a:=float(C/d^2);
    ax:=-a*sx[i]/d; // sx[i]/d ist die cos-Komponente incl. VZ-Änderung
    ay:=-a*sy[i]/d; // sy[i]/d ist die sin-Komponente
    sx[i+1]:=sx[i]+vx*dt+0.5*ax*dt^2;
    sy[i+1]:=sy[i]+vy*dt+0.5*ay*dt^2;
    vx:=vx+ax*dt;  // neue Geschwindigkeit am Ende von dt
    vy:=vy+ay*dt;
  end_for;
  end_proc:
  ```

- ```
  dt:=60:; // Dauer einer Zeiteinheit in Sekunden
  vy0:=29880:; //Anfangsgeschw. im Aphel im m/s
  vy0:=20000:;
  vx0:=0:;
  sx0:=1.5*10^11:;  // Abstand Erde Sonne in m
  sy0:=0:;
  n:=365*24*60:; // Anzahl der Minuten im Jahr
  Bahn(sx0,sy0,vx0,vy0,n,dt):;
  ```

- ```
  //k:=round(24*60*60/dt):; // Plotten im 24-Std-Abstand
  k:=round(7*24*60*60/dt):; // Plotten im Wochen-Abstand
  k_max:=n div k:;
  Punkte:= plot::Point2d([sx[k*i],sy[k*i]],
                             PointSize=2,
                             PointStyle=FilledCircles,
                             Color=RGB::Red,
                             VisibleFromTo = i-1..i+1)
                             $ i=1..k_max:;

  Bahn:=plot::Line2d([sx[k*i],sy[k*i]],[sx[k*(i+1)],sy[k*(i+1)]],
                             LineColor = RGB::Red,
                             VisibleFromTo = i..i+1,
                             VisibleAfterEnd)
                             $ i=0..k_max-1:;

  Strahlen:=plot::Line2d([sx[k*i],sy[k*i]],[0,0],
                             LineColor = RGB::Red,
                             VisibleFromTo = i-1..i+1)
                             $ i=1..k_max:;

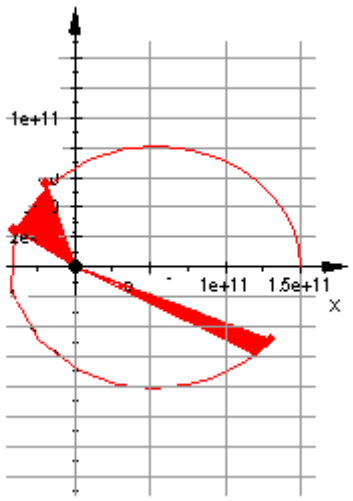
  Sonne:=plot::Circle2d(6.96*10^8,[0,0],LineColor=RGB::Black,
      FillPattern=DiagonalLines,FillColor=RGB::Black,Filled=TRUE):;
  ```
  ```

- ```
  b1:=1.6*10^11:;b2:=trunc(0.95*b1):; // ca. Erdbahnradius
  plot(Sonne,Punkte,Bahn,Strahlen,
       ViewingBox = [-b1..b1, -b2..b2],
       GridVisible = TRUE):;
  ```

Der Fahrstrahl überstreicht in gleichen Zeiten gleiche Flächen:

- ```
  Flaeche:=proc(x1,y1,x2,y2)
  local d1,d2,d,alpha;
  begin
    d1:=sqrt(x1^2+y1^2);
    d2:=sqrt(x2^2+y2^2);
    alpha:=float(abs(arccos(x1/d1)-arccos(x2/d2)));
    d:=float(0.5*(d1+d2)):; // Mittelwert der Entfernungen
    A:=float(0.5*alpha*d^2); // Kreisaussschnitt mit Mittelwert als Radius
  end_proc:;
  ```

- ```
  k_max;
  DIGITS:=4:;
  for i from 0 to k_max-1 do
    Flaeche(sx[i*k],sy[i*k],sx[k*(i+1)],sy[k*(i+1)]);
    print(NoNL,Unquoted,A,"  ");
end_for:;
  ```




```
9.060e20,   9.060e20,   9.060e20,   9.061e20,   9.061e20,   9.063e20,
9.065e20,   9.068e20,   9.076e20,   9.091e20,   9.124e20,   9.202e20,
9.372e20,   1.859e20,   9.417e20,   9.229e20,   9.136e20,   9.097e20,
9.079e20,   9.071e20,   9.067e20,   9.064e20,   9.063e20,   9.062e20,
9.062e20,   9.062e20,   9.062e20,   5.192e20,   9.062e20,   9.062e20,
9.062e20,   9.063e20,   9.064e20,   9.066e20,   9.069e20,   9.075e20,
9.088e20,   9.115e20,   9.180e20,   9.327e20,   6.033e20,   9.464e20,
9.264e20,   9.152e20,   9.104e20,   9.083e20,   9.074e20,   9.069e20,
9.066e20,   9.065e20,   9.064e20,   9.063e20
```
```