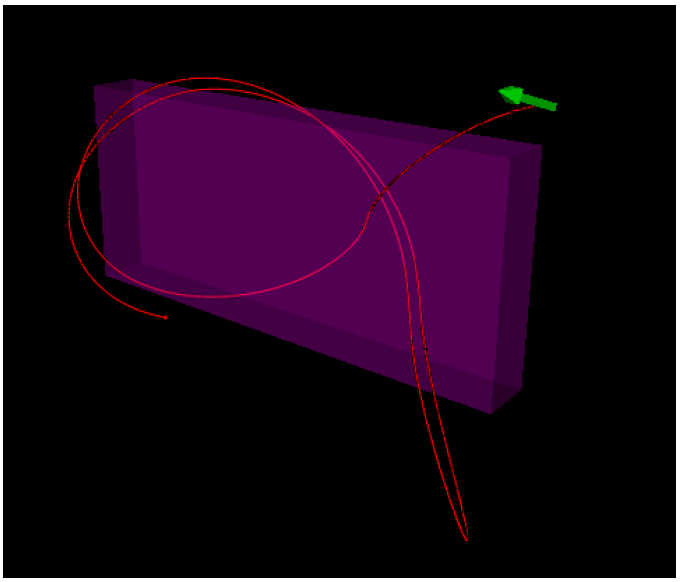
# VPython Simulation of a Right Rectangular Gravitational  Prism

*Kari Peisa, 2019*



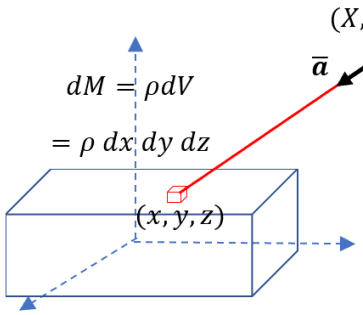
This document presents the mathematical grounds for the VPython simulation in a gravitational field produced by a rectangular monolith. It also discusses some implementation problems when programming the math in Python.

The context of the simulation is scientific fiction. In this simulation the gravitational prism is a space volume of  “dark matter” which produces the interaction of gravity, but it has no rigid body behaving i.e. in the simulation, the spaceship can pass through the prism. The gravitational field outside and inside of the prism is produced according to the Newton’s law of universal gravitation.

The following documents have inspired the simulation to be completed:
(1) Stanislav Barton, Veikko Keränen: Clarken Avaruusseikkailu ja Newtonin gravitaatiolaki (Finnish)
https://www.dimensiolehti.fi/clarken-avaruusseikkailu-ja-newtonin-gravitaatiolaki/
(2) Buddhader Banerjee and P. Das Gupta: Gravitational Attraction of a Rectangular Parallepiped
Geophysics, vol. 42, No. 5 (August 1977); P. 1053-1055.
(3) James M. Chappell, Azhar Iqbal, and Derek Abbott: The gravitational field of a cube
https://arxiv.org/pdf/1206.3857.pdf

# Newton's Gravitation Law

Figure 1.




$$d\bar{\boldsymbol{a}}(X,Y,Z) = -\kappa \frac{dM}{R^2}\bar{\boldsymbol{R}}^0 = -\kappa\rho \frac{dV}{R^2}\bar{\boldsymbol{R}}^0$$

$$\bar{\boldsymbol{a}}(X,Y,Z) = -\kappa\rho \iiint_V \frac{dx\, dy\, dz}{R^2}\bar{\boldsymbol{R}}^0$$

Monolith is a right rectangular prism that the spaceship is approaching. According to Arthur C. Clarke's A Space Odyssey the monolith's sides were on a ratio 1:4:9 which is used also in this simulation.

The equation for universal gravitation takes the form

$$\boldsymbol{F} = -\frac{\kappa m_1 m_2}{r_{12}^2}\boldsymbol{r}_{12}^0$$

where

$\kappa$=Gravitational constant =6.6723 $10^{-11}$ $m^3$ $s^{-2}$ $kg^{-1}$

$\boldsymbol{r}_{12}^0 = \frac{(X-x, Y-y, Z-z)}{\sqrt{(x-X)^2+(y-Y)^2+(z-Z)^2}}$ is the unit distance vector from mass $m_1$ to mass $m_2$

Usually, the other mass is much heavier than the other, and we can assume that it does not move. The acceleration of the lighter mass at some space point in the gravitational field of the heavier mass is obtained from Newton's law by dividing the the lighter mass. If the masses in Newton's law are homogeneous spheres, we can assume that their mass is centered. This is shown, for example, in document (1). However, with a rectangular monolith this assumption does not hold.

As it is presented in the Figure 1., the acceleration is produced by the total mass of all differential volume units of the monolith. Since the distance between the units and the spaceship varies, we have to integrate over the whole volume. This is very interesting mathematical and programming problem for our simulation.

Let us set:
The center of the monolith is in origin, and the space ship position is in (X,Y,Z).
The ratio 1:4:9 of the sides of the monoliths yields the limits of the integration x1=-9/2T, x2=9/2T, y1=-2T, y1=2T, z1=-1/2T, and z2=1/2T, where T is the length of the shortest side.

Now, we can determine the acceleration at space point (X,Y,Z) by equation 1.

$$\boldsymbol{a} = (a_x, a_y, a_z) = \kappa\rho \int_{-\frac{T}{2}}^{\frac{T}{2}}\left(\int_{-2T}^{2T}\left(\int_{-\frac{9}{2}T}^{\frac{9}{2}T} \frac{(x-X,\ y-Y,\ z-Z)}{R^3}dx\right)dy\right)dz \qquad (1)$$

where $R = \sqrt{(x-X)^2 + (y-Y)^2 + (z-Z)^2}$ . The integration is applied to each component separately.

The order of the integration variables can be selected freely.

We can compute the result of the definite triple integral in many ways. Below we present tree ones.

**First**, in mathematical software obeying symbolic computation it is possible to compute definite triple integrals just by writing them. In many cases, like here, it can take a huge amount of time, or it is not possible at all. In VPython, we can not use these environment.

**Second**, we can compute or derive on a paper the analytic solution of the integral function **F**(x,y,z). After that we can use symbolic substitutions to get the result

$$\boldsymbol{a} = \kappa \rho \left[ \left[ \left[ \boldsymbol{F}(x, y, z) \right] \right]_{x=x1}^{x2} \right]_{y=y1}^{y2} \Big]_{z=z1}^{z2} \qquad (2)$$

In Python programming there is module SymPy for symbolic mathematics and computer algebra. Module SymPy was tested to compute the substitutions of the triple integral (2). The module managed, but unfortunately the rewriting processes needed in symbolic substitutions proved to be too slow for simulation purposes.

There is also module SciPy which is a collection of mathematical algorithms and convenience functions including also integration function quad(). This module was not tested, since the third solution was found.

**Third**, we can compute the result of substitutions numerically by the following decomposition:

$$\int_{z1}^{z2}\int_{y1}^{y2}\int_{x1}^{x2} \mathbf{f[x,\,y,\,z]}\,\mathbf{dx}\,\mathbf{dy}\,\mathbf{dz} = \int_{z1}^{z2}\int_{y1}^{y2} \left(\mathbf{int1[x2,\,y,\,z]} - \mathbf{int1[x1,\,y,\,z]}\right)\mathbf{dy}\,\mathbf{dz} =$$

$$\int_{z1}^{z2} \left(\mathbf{int2[x2,\,y2,\,z]} - \mathbf{int2[x2,\,y1,\,z]} - \left(\mathbf{int2[x1,\,y2,\,z]} - \mathbf{int2[x1,\,y1,\,z]}\right)\right)\mathbf{dz} =$$

$$\mathbf{F[x2,\,y2,\,z2]} - \mathbf{F[x2,\,y2,\,z1]} - \left(\mathbf{F[x2,\,y1,\,z2]} - \mathbf{F[x2,\,y1,\,z1]}\right) -$$
$$\left(\mathbf{F[x1,\,y2,\,z2]} - \mathbf{F[x1,\,y2,\,z1]} - \left(\mathbf{F[x1,\,y1,\,z2]} - \mathbf{F[x1,\,y1,\,z1]}\right)\right) =$$

$$\left(\mathbf{F[x2,\,y2,\,z2]} - \mathbf{F[x2,\,y2,\,z1]}\right) - \left(\mathbf{F[x2,\,y1,\,z2]} - \mathbf{F[x2,\,y1,\,z1]}\right) -$$
$$\left(\mathbf{F[x1,\,y2,\,z2]} - \mathbf{F[x1,\,y2,\,z1]}\right) + \left(\mathbf{F[x1,\,y1,\,z2]} - \mathbf{F[x1,\,y1,\,z1]}\right)$$

The final substitutions can be computed in a normal python function, and it proved to be fast enough for simulation. Besides, no extra modules needed to be loaded.

## The analytic solution of the triple integral

Let us explore the analytic solution to triple integration for the first component of acceleration in origin. The solution at point $(X, Y, Z)$ is obtained simply by the substitutions $x \to x - X$, $y \to y - Y$, and $z \to z - Z$.

The first two integrations are found easily by basic integration methods. But the final integral function needs more complicated methods. With Mathematica's algorithms we get the following solution:

In[ ]:=
```
int1 = Integrate[x / (Sqrt[x^2 + y^2 + z^2])^3, x];
int2 = Integrate[int1, y]
int2 // FullSimplify
```

Out[ ]=
$$\frac{1}{2}\,\text{Log}\left[1 - \frac{y}{\sqrt{x^2+y^2+z^2}}\right] - \frac{1}{2}\,\text{Log}\left[1 + \frac{y}{\sqrt{x^2+y^2+z^2}}\right]$$

Out[ ]=
$$-\text{ArcTanh}\left[\frac{y}{\sqrt{x^2+y^2+z^2}}\right]$$

In[ ]:=
```
int3 = Integrate[int2, z]
```

Out[ ]=
$$x\,\text{ArcTan}\left[\frac{y\,z}{x\sqrt{x^2+y^2+z^2}}\right] + \frac{1}{2}\,z\,\text{Log}\left[1 - \frac{y}{\sqrt{x^2+y^2+z^2}}\right] - \frac{1}{2}\,z\,\text{Log}\left[1 + \frac{y}{\sqrt{x^2+y^2+z^2}}\right] - y\,\text{Log}\left[z + \sqrt{x^2+y^2+z^2}\right]$$

The last form can be transformed to the final analytic solution of integral function of the form:

```
F[x_, y_, z_] :=
```
$$x\,\text{ArcTan}\left[\frac{y\,z}{x\sqrt{x^2+y^2+z^2}}\right] - \frac{1}{2}\,z\,\text{Log}\left[\frac{y+\sqrt{x^2+y^2+z^2}}{-y+\sqrt{x^2+y^2+z^2}}\right] - y\,\text{Log}\left[z+\sqrt{x^2+y^2+z^2}\right]$$