PLANETARIUM SIMULATION SYSTEM

Prof. Dr. W. Kowalk
FB Informatik – Universität Oldenburg, Germany
Ammerländer Heerstraße 114-128, D-26129 Oldenburg
Kowalk@informatik.uni-oldenburg.de

Abstract

This paper describes a modeling, simulation, and evaluation system for planetary simulation, including dynamic 3D-visualization, event capturing, and experiment management.

Because of the amount of information to be given, particular presentation of detailed pictures, we plan to inform in this paper about main aspects of our simulation system and in a following paper about handling of and experimenting with this system.

Keywords

MODELLING AND SIMULATION METHODOLOGIES, 3-Dimensional Modeling, Visualization, Applications

1 Introduction

Simulation of real world systems has been a long tradition, where accuracy of the results is usually the main quality feature. Although we agree to this statement, there are further important requirements for simulation systems, where visualization of an evolving simulation to convince modelers as well as customers from its plausibility, and event capturing to protocol particular incidents are only two. The simulation tool that is presented here considers several requirements for continuous time simulation systems.

We call our system **Copernicus Planetarium** in respect to a (well known) former Physician, whose problems were not only of mathematical or technical nature, but have been political as well as religious; this is to remind people of our time that progress has not only been achieved in mathematics or computer technique, but also in social tolerance and human respect.

Our system is programmed completely in Java, which has of course some drawbacks according to performance. However, on a modern computer system we can evaluate several million steps in a second, which seems to be sufficient for many applications, as will be shown later.

Our paper starts with some technical remarks on simulation techniques, which are in general not very new, although some problems like event time calculation may be of particular interest. Another feature is the presentation technique, which displays the movement of planets in an animated space, that can be switched to several display methods, including tracing of planet courses, three-dimensional presentation by use of red-green-glasses or different viewing perspectives. Then we will present the organizational background of our simulation system, and as a final technical discussion we are concerned with event capturing, i.e. how to protocol events like “planet arrives at the nearest point to sun (perihelion)” or “two planets crashes”, the latter of which is of interest in chaotic modeling.

In the second course of this paper we present some simulation results, which demonstrates the high accuracy of the applied technique as well as the high educational effect, so that this system might be useful in education as well as astronomical planetoid simulation.

2 Technical Considerations

We start with some technical remarks on simulation techniques used in this simulation system. We developed a polynomial extrapolation technique, which has been evaluated to be very accurate. We also introduce some new methods of computing special events like arriving at the minimum or maximum distance to the central star or crashing of two planets.

2.1 Mathematical Methods

This section describes the mathematical background for the simulation system. We use a well known polynomial approximation technique, however, use also some more evaluated methods to compute exact co-ordinates for perihelion (nearest distance to sun), aphelion (farthest distance from the sun), or co-ordinate axes crossing.

Our approach is clearly a multiple step method, similar to the well known Adams-Bashforth method [e.g. 1], where we extrapolate from some points in past the next point in future. Instead of analyzing differential equations we focus to the more direct approach that applies gravitational law alone to the planets and compute from an estimated average of force (and therefore acceleration) a mean for acceleration, speed and finally position. This method can also be applied to correction by re-estimating extrapolated quantities.

Our simulation system uses three dimensional coordinates, which we call as usual x, y and z. For each point $s=(x,y,z)$ we need the speed of the body $v=(v_x,v_y,v_z)$ and the acceleration $a=(a_x,a_y,a_z)$ in the corresponding points. We com-

pute and remember those values at three points in sequence, which we call s_0, s_1, s_2, v_0, v_1, v_2 and a_0, a_1, a_2, respectively. Given those points, we extrapolate the average speed in the next step (between v_2 and v_3) by a formula described below and estimate the next point s_3. In this point, acceleration is computed by gravitational forces in this point, from which the average acceleration between a_2 and a_3 is used to compute the speed in v_3. Now, all indices are shifted ($s_0=s_1$, $s_1=s_2$, $s_2=s_3$ etc.) and the next step can be computed.

There are two formulae required to compute average values, one is to extrapolate between v_2 and v_3 from (v_0, v_1, v_2), the other to interpolate between a_2 and a_3 from (a_1, a_2, a_3), where a_3 is estimated by the physical model. To determine a polynomial of degree *2* we use the well known interpolation formula from Lagrange

$$f(x)=\frac{(x-x_1)\cdot(x-x_2)}{(x_0-x_1)\cdot(x_0-x_2)}\cdot u_0+\frac{(x-x_0)\cdot(x-x_2)}{(x_1-x_0)\cdot(x_1-x_2)}\cdot u_1+ \\ +\frac{(x-x_0)\cdot(x-x_1)}{(x_2-x_0)\cdot(x_2-x_1)}\cdot u_2.$$

To get $f(-\Delta)=u_0$, $f(0)=u_1$, $f(+\Delta)=u_2$ with given $x_0=-\Delta$, $x_1=0$, $x_2=\Delta$, we get the formula:

$$f(x)=\frac{x^2}{2\cdot\Delta^2}\cdot(u_0-2\cdot u_1+u_2)+\frac{x}{2\cdot\Delta}\cdot(u_2-u_0)+u_1.$$

We can extrapolate this function easily, by setting $x=2\cdot\Delta$:

$$f(2\cdot\Delta)=2\cdot(u_0-2\cdot u_1+u_2)+u_2-u_0+u_1= \\ =u_0-3\cdot u_1+3\cdot u_2=u_0+3\cdot(u_2-u_1)$$

which is very plausible, since in linear case the outcome is exactly a straight line, or if $u_2=u_1$, the result is u_0. To be more formal, if f where the real function of degree *2*, its derivatives are

$$f'(x)=\frac{x}{\Delta^2}\cdot(u_0-2\cdot u_1+u_2)+\frac{1}{2\cdot\Delta}\cdot(u_2-u_0),\ f''(x)= \\ =\frac{1}{\Delta^2}\cdot(u_0-2\cdot u_1+u_2),$$

and its derivatives at Δ are:

$$f'(\Delta)=\frac{1}{\Delta}\cdot(u_0-2\cdot u_1+u_2)+\frac{1}{2\cdot\Delta}\cdot(u_2-u_0)= \\ =\frac{1}{2\cdot\Delta}\cdot(u_0-4\cdot u_1+3\cdot u_2), \\ f''(\Delta)=\frac{1}{\Delta^2}\cdot(u_0-2\cdot u_1+u_2).$$

Now, using Taylor extension

$$f(x)=f(\Delta)+f'(\Delta)\frac{x-\Delta}{1!}+f''(\Delta)\frac{(x-\Delta)^2}{2!}+R(x)= \\ =\frac{x^2}{2\cdot\Delta^2}\cdot(u_0-2\cdot u_1+u_2)+\frac{x}{2\cdot\Delta}\cdot(u_2-u_0)+u_1.$$

we get the same result for f as from above. This demonstrates that our method is mathematically the same as known multi-step methods, where we now explicitly apply an approximation function to several measured points instead of derivatives in one point.

Integrating the function *f* from *0* to $+\Delta$, we get

$$\int_0^{\Delta} f(x)\,dx=\frac{\Delta}{12}\cdot(-u_0+8\cdot u_1+5\cdot u_2).$$

whereas integrating from $+\Delta$ to $+2\Delta$, we get

$$\int_0^{2\cdot\Delta} f(x)\,dx=\frac{\Delta}{12}\cdot(5\cdot u_0-16\cdot u_1+23\cdot u_2).$$

We find a set of coefficients that can be used to compute integrals from interpolated (-1/12, 8/12, 5/12) and extrapolated (5/12, -16/12, 23/12) sequences of values, which have been proven to be very accurate. Also, we find some sequences of points that can be used to find extreme points, like minimum or maximum. Let us assume, that the distance $d_k=|s_k|$ is computed by the usual Euclidian formula. At those three points we have three distances (d_0, d_1, d_2) to the sun, where we find a minimum when $d_1<d_0$ and $d_1\leq d_2$. Thus there must be a minimum between d_0 and d_2. Setting $f(s)=|s|$ from above and differentiate we get:

$$f'(t)=\frac{2\cdot t}{2\cdot\Delta^2}\cdot(u_0-2\cdot u_1+u_2)+\frac{1}{2\cdot\Delta}\cdot(u_2-u_0)= \\ =\frac{t}{\Delta^2}\cdot(u_0-2\cdot u_1+u_2)+\frac{u_2-u_0}{2\cdot\Delta}=0$$

or

$$t_{min,mix}=\frac{\Delta}{2}\cdot\frac{u_0-u_2}{u_0-2\cdot u_1+u-2}.$$

We find a relatively simple approximative formula to compute the time t_{min}, when minimum distance to sun is achieved, which seems not to suffer from numerical problems; in experiments this formula has been proved to be very exact. From this the coordinates can be computed by the formula $f(t_{min})$. The same formula holds for maximum distance.

Now, let us assume that we have for three points at time $-\Delta$, *0* and $+\Delta$:

$$f(-\Delta)=u_0<f(0)=u_1\leq 0\leq f(-\Delta)=u_2,$$

and we want to estimate the time, when $f(t)=0$. If *f* is a polynomial of degree *2*, we can solve this simply by using the corresponding algebraic square root formula. However, those formulae have been proved to be numerically very unstable, so that another solution had to be found. We used an approximation function, from Newton, which yields:

$$t=t-\frac{a\cdot t^2+b\cdot t+c}{2\cdot a\cdot t+b}=\frac{a\cdot t^2-c}{2\cdot a\cdot t+b}$$

and resulted in numerically very stable results.

We have tried to improve some of those formulae. For example, approximating s_3 from extrapolated values of $v_{ext}=(v_0, v_1, v_2)$ might lead to inaccurate results, so that we re-computed from estimated value of $s'_3=s_2+v_{ext}\cdot\Delta$, acceleration in a'_3, from this average speed in $v_{av}=(a_1, a_2, a'_3)$, and now the new value for $s_3=v_{av}\cdot\Delta$. However, the differences were hardly observable, so that the final system does not use this corrective iteration.

Another problem occurs when simulation is to be initiated.

In many cases little problems occur with rough estimations of those initial values, since after some iterations the results become considerably accurate. But when simulation starts at a critical point, initial values must be carefully considered. This time we use an iterative method, i.e. estimate some approximative values for the positions, compute accelerations and velocities in those points, and then re-estimate the positions. These steps are repeated several times using the re-estimated values, yielding satisfying results.

Finally, one might think that higher approximations might be more accurate. They are simply introduced here by using the corresponding Lagrange formula of degree three:

$$f(x)=\frac{(x-x_1)\cdot(x-x_2)\cdot(x-x_3)}{(x_0-x_1)\cdot(x_0-x_2)\cdot(x_0-x_3)}\cdot u_0+\frac{(x-x_0)\cdot(x-x_2)\cdot(x-x_3)}{(x_1-x_0)\cdot(x_1-x_2)\cdot(x_1-x_3)}\cdot u_1$$
$$\frac{(x-x_0)\cdot(x-x_1)\cdot(x-x_3)}{(x_2-x_0)\cdot(x_2-x_1)\cdot(x_2-x_3)}\cdot u_2+\frac{(x-x_0)\cdot(x-x_1)\cdot(x-x_2)}{(x_3-x_0)\cdot(x_3-x_1)\cdot(x_3-x_2)}\cdot u_3$$

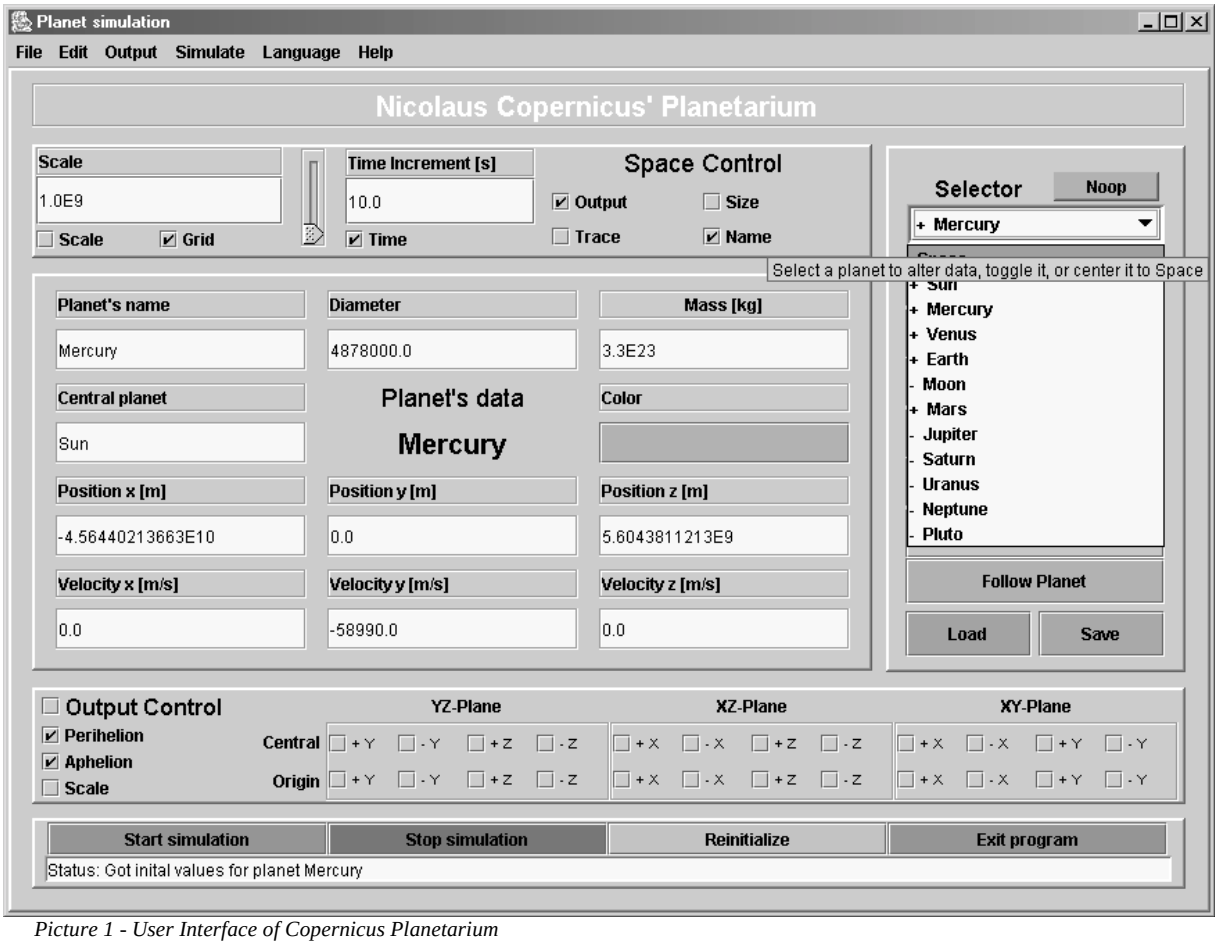
and continue as above. The results have not been found to be more accurate than those from above. As will be presented in the second part of this paper, our results using second order approximation are very precise. Errors might be introduced by rounding of intermediate results, which influences the results much more than the order of the approximation function.

2.2 Presentation Methods

This section describes the presentation methods of our simulation system. If you start the program there appears a panel, which controls all features of the simulation. When a simulation is started, another window pops up which is called “Space”, and in which the observer can follow the course of the planets. Further information can be displayed in text windows, the contents of which can be stored in files. Additionally, some intermediate quantities can be displayed on the main panel. Several output windows can be maintained during simulation, so that different results can easily be compared. These features will now be considered in more detail.

Main Panel

The main window controls all further activities. It holds the planets to be simulated in a drop down menu, controls output to the “space”, where the planets course can be displayed, and controls several parameters of the planets. The figure shows this panel, with planet Mercury and some of its parameters selected. The panel on the right hand side controls the flow of data between the planet data base and the planet's data panel. Also, new planets can be added, or existing planets removed. and file input and output controlled.



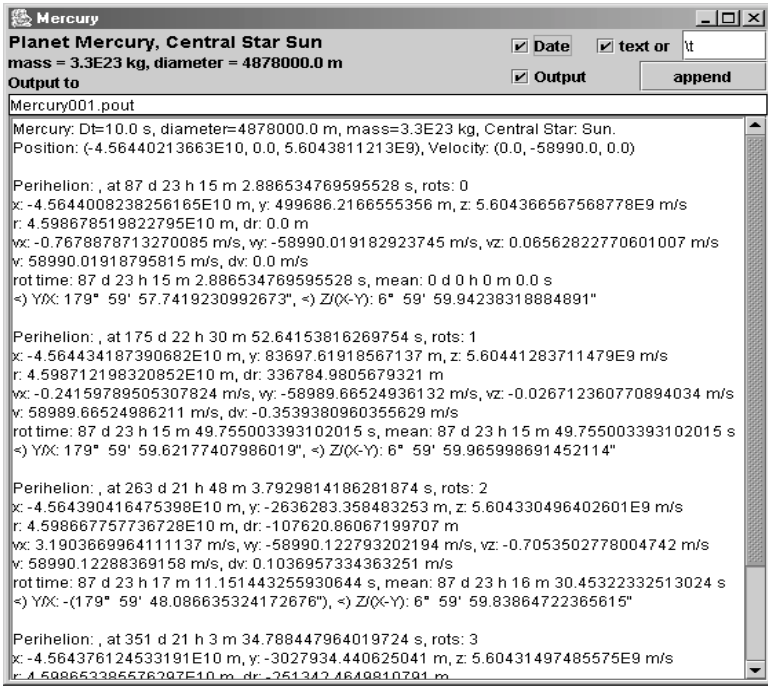
Picture 1 - User Interface of Copernicus Planetarium

In the upper panel you find some check boxes, input fields, and sliders to control the space-output. In the lower field output control of events can be specified. Simulation is started, interrupted, or stopped by some buttons in the lower field. There is also a status bar, that is used to display actual events. Although, the color of a control panel is usual of minor interest, our system offers the possibility to control the color of all buttons of the system. This is done to "customize" the system to each users personal preferences, which helps them to become as familiar as possible with the interface. Of course, internationalization is provided as well, currently in English and German; precaution is given to localize this to any other language, when the corresponding resource files are supplied.

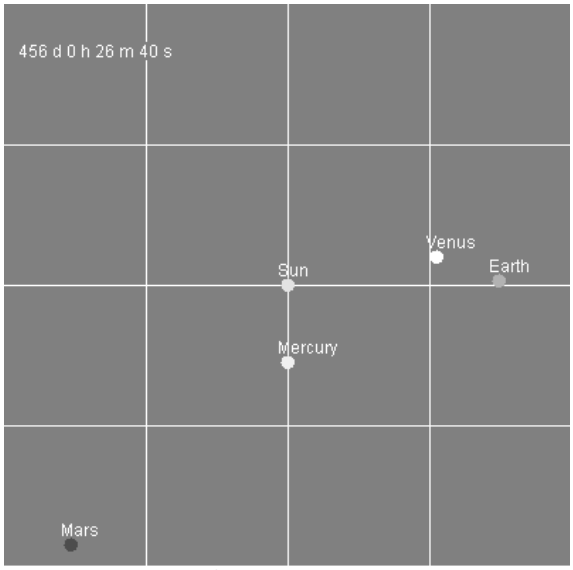
Space

There is one graphical window (called ***Space***), that is used to display the animated position of the planets. This is very important for users, since they can easily see whether a simulation run meets their major requirements. Also it demonstrates the results to other viewers in a much more obvious way than by lists of numbers.

Animation is displayed in a window called ***Space***. Here, all planets can be displayed, as well as their current position. It is possible to increase this window arbitrarily, however, because of current speed limits of the Java engine, it is usually better to keep this window small. However, the window can be scaled to each size, so that any extent can be displayed in one panel. Since the distance from the sun varies for all planets very much, this is an important feature. Other parameters are time, scale, or even background color, but also grid or view. View can be changed by several parameters. For example, since we use three dimensional spatial simulation, simulation can be observed from the horizontal point of view, observing the bending from the ecliptic, the area, in which the earth rotates around the sun. Particular comets like Haley have a very large bending to the ecliptic, however Mercury's or the Earth's Moon's angle to the ecliptic is also considerably large.



Picture 2 - A Text Window displays exact values during simulation



Picture 3 - Output of a simulation run with Sun and 4 planets

Sometimes, the trail of a planet is of interest. Observing an ellipse is of course of some interest, however, our simulation system allows to specify any planet as "center of observation". Thus, instead of having the origin of the space as center, the Sun, the Earth or any other planet can become the center by pressing the button "Follow Planet". This leads (e.g. in case of geocentric view) to very complicated trails, and it's of interest to display these trails. This can be done by clicking the "Trace" check box. Also, the original diameter of a planet can be displayed, although in most cases this is very small compared to the distances between planets and sun. Anyway, the figure that come out here, are very beautiful.

Animated planet movement or displaying the trail of a planet leads to very interesting results, which should help to understand such physical phenomena. Thus, this helps to understand astronomical results, which is useful for teaching and researching.

Text windows

Many exact results can be displayed in special windows, which are opened automatically, when for example Output Control is selected. To be more concrete, look at Picture 3; let us assume we want to observe the shift of the perihelion (smallest distance to the sun) of planet Mercury. Mercury's trail should be very stable in case of a two-body simulation, i.e. there is no influence of other planets or relativistic effects. To do so, for planet Mercury

Output Control is to be selected, as well as perihelion. Starting simulation opens besides Space another text window, which is here used to show the parameters at the moment when Mercury's distance to sun becomes a minimum. Besides others, also the distance and the angle of the corresponding point is displayed on the text panel. It is possible to transfer this to a spread sheet, where the parameters can be displayed in a diagram. This helps to display scientific relevant results. For example, the following picture shows the shift of the angle of planet Mercury, if besides the Sun also the planet Venus is present. For each planet such a window can be opened, where for all planets distinct events can be defined, which produce an output to its dedicated window. Besides perihelion and aphelion, also crossing of X-Y-plane etc. can be defined as events.

Another text window can be defined, which lists all outputs to the status bar. An event which can here be displayed is collision of two planets.

2.3 Event Capturing and Further Options

We have shown in the last section that simulation can be stopped by pressing a mouse button. However, simulation can also stop when some events occurs, particularly, when two planets crash (i.e. their distance is smaller than the sum of their radii). Thus such “cosmic events” can easily be observed – and after this the simulation might be continued. In these and other cases it is possible to protocol events in another text panel, the content of which can be stored to a file, either automatically or by a mouse click.

The format of the text panel can be altered by a mouse click to assist evaluation of the data in a spread sheet (e.g. semicolon separated). This simplifies analysis of simulation runs, even over very long period, demonstrating accuracy of the simulation system itself, or some physically relevant effects like the shift of Mercury's perihelion. This will be demonstrated in the second part of this paper.

The simulation system allows several options according to the physics applied. The feature “Central star fix”, allows that the first star of the planets list is not moved. The feature “Central gravitation only” allows to select, that only gravitational forces from the central star are computed. This makes simulation much faster, however, also less accurate.

3 Performance of the Simulation System

3.1 Relative accuracy of simulation runs

To show the accuracy of our simulation algorithm we measured the perihelion of planet Mercury after several revolutions (altogether 1200) and estimated the differences to the first position. Picture 4 shows the result. The differences of the distances at the perihelion are usually less than 13 cm; compared to the total distance of 46 Million km this is negligible. Also, the difference of the angle (which is measured in one Million degree second, to scale it to visibility) is very stable. Thus, this simulation system confirms the theoretical well known result, that the ellipse in which a planet runs is very stable in space. Of course, in real world systems differences are much greater, since there are for example influences of other planets to the planet; or the “dust” in the space, e.g. Sun winds or meteors, alter the planet's course. The latter is not considered in our simulation system, however, the first one is.

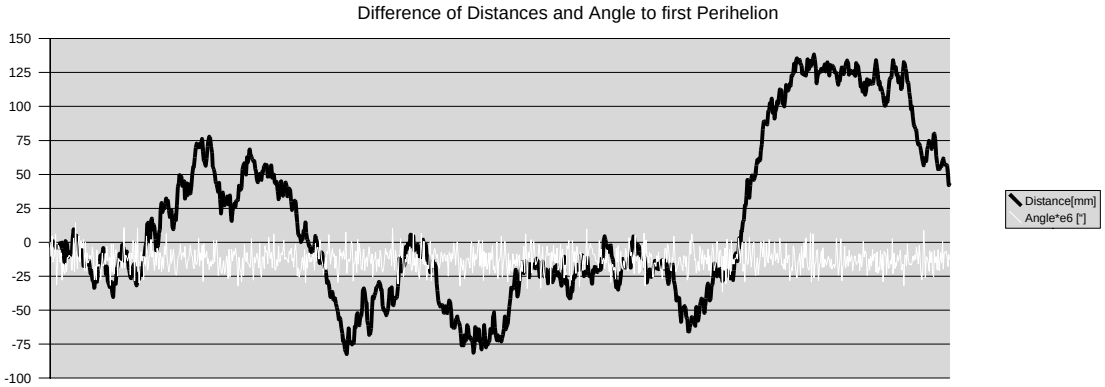
Influences of other planets can easily be introduced, by simply simulate the system with several planets and measure and display the results. Picture 5 shows the corresponding results after measurement of more than 1000 (exactly 1036) revolutions of Mercury around Sun, including Venus. While the distance (here in km) jitters around zero, the angle (here in seconds) drifts monotonously to increasing values, which shows that Venus (and of course also other planets) influence the orientation of Mercury's ellipse. The maximum jitter in distance has been measured to be about 900 km, which is compared to Mercury's diameter of about 4878 km a considerable amplitude. The drifting of the angle seems to converge to about 50” per 100 revolutions of Mercury around the Sun.

These results show on one side the accuracy of our simulation systems, which simulated much more precisely in the first case than planets actually behave. However, it shows also in the second case, that in real world system many influences to a phenomenon exist, which cannot all be considered adequately, so that any result must always be checked against real world observation. Our simulation tool might help to understand many effects much better, but it can never substitute observation of real world system.

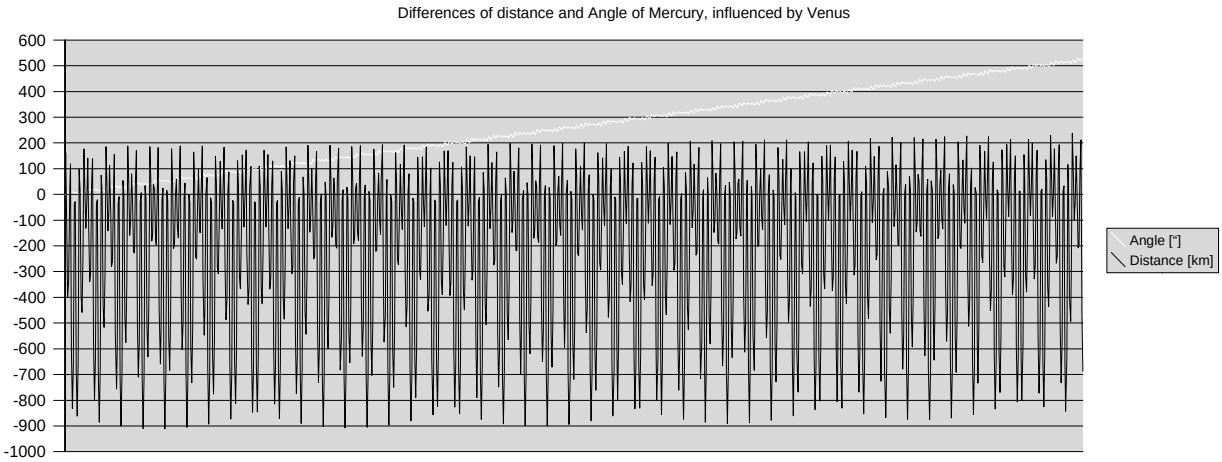
As another example we measure the revolution time of planet Venus and its (average) distance to Sun. The relationship (Kepler's 3. Law [2]) from which we get for the distance r=108202203533.893 m the time 224 d, 16 h, 26 m, 59.47 s. Our measurement yields 224 d, 16 h, 26 m, 35.69 s, which is very close to this result. The difference might come from the inaccuracy of the physical constants like the gravitational constant $\gamma = 6.672 \cdot 10^{-11}$. Again we see that our simulation systems produces very precise absolute results.

3.2 Absolute accuracy of simulation runs

Another test shows the absolute accuracy of our system, which is – compared to the relative accuracy as in the last example – relevant for measurement of planetary movements. In this first example we measured the escape velocity of a body by dropping them to the Sun or Earth. The model is simple: Simulate two non-moving bodies in a relative large distant, where one of them has very big mass compared to the other (e.g. Sun – Mercury), and as soon as Mercury approaches to the Sun more than the radius of the Sun, measure Mercury's velocity. The result has been 617449.2 m/s, where the theoretical result is 617.4 km/s (the latter cannot be computed more exactly, since the physical constants are not known with higher precision [Keller94]). This shows, that our simulation system can be used also to estimate observable physical quantities with good quality.



Picture 4 - Accuracy of simulation runs: Difference of Distance and Angle to first Perihelion.



Picture 5 - Drift of a planet's perihelion angle by another planet: Differences of Distance and Angle of Mercury, influenced by Venus.

3.3 Efficiency of the simulation system

This Simulation system has been implemented in Java to support portability, and also to explore Java's ability to be used as a programming language in an area where high execution performance is required. So some figures will be given here. Running a 1 GHz-Pentium 3 Laptop (DELL Latitude C800), we simulated one planet (always plus Sun) with a rate of ca. 475,000 steps per second. E.g. in 1 minute using a step width of 10 s model time we simulated 3290 days model time. Simulating 4 planets, where each planet attracts each other, we achieved a simulated model time of 805 days in 1 minute real time, or more than two Earth years. 10 s step time has been used in all experiments results of which are presented here.

Output of animated movement of planets to the space window can be inhibited or refresh time of space window can be manipulated, which improve simulation time.

4 Conclusions

This paper has presented a simulation system, capable to demonstrate the behavior of galactic objects like planets, meteors, or satellites. It uses a high precision simulation technique that yields much more precise results than required for realistic simulation, since other perturbations disturb the course of a planet much more than the highly accurate simulation suggests.

Besides high precision also usability of the software had been taken care for. Our system allows to store and access single objects, as well as experiments, or complete simulation runs. Results can be stored in files, or formatted so that the data can be manipulated in spread sheets; our diagrams have all been produced in this way. This will be more elaborated on in a following study.

There is a tutorial in progress (currently only in German), which shows that this simulation system can be used for educational purposes, as well. Since internationalization is considered (which will also sustain several additional important languages in near future) our system might be used for these applications as well. For this, the simulation system can be purchased for a very low price.

A second paper [3] that treats of this planetarium simulation system will demonstrate its usability by performing some simulation experiments and presenting the results.

5 References

[1] J. Banks, J. S. Carson II, B. L. Nelson: *Discrete-Event System Simulation.* Prentice Hall 1996.

[2] H..-U. Keller: Astrowissen: *Zahlen – Daten – Fakten.* Frankh-Kosmos Verlag Stuttgart 1994.

[3] W. Kowalk: *Planetarium Simulation System – Usability and Experiments.* To be published.