Time step and time scale

A **time step** is an increment of the time variable (dt, delta_t, Δt) that you use in your programs to update position and momentum. This is an inherent feature of computer simulations: none of the physical quantities can really be made continuous in time, and one has to perform calculations in discrete steps.

The questions I would like to address here are: How to choose the time step correctly in your program? Is it always a good choice to set it automatically to 0.01 s or 0.001 s? What determines the time step in general?

A simple answer is that the time step has to be small. But small compared to what? And here we arrive at the notion of a characteristic time scale of a problem.

A **time scale** is defined as a typical or characteristic time set by the physical processes in a given physical problem. For example, what is the time scale for the Moon orbiting the Earth? In this case, a characteristic time is the time of one revolution, or the period, which is equal to about 1 month. On the other hand, for an electron orbiting a proton in a hydrogen atom, a characteristic time scale is a tiny fraction of a second, of the order of $10^{-16}$ to $10^{-15}$ s. Does it make much sense to select a time step of 0.01 s in the first case, when one revolution takes 1 month, or about 260 million times 0.01 s? No, of course not, this is a way too small time step, which is not really needed to capture the physics correctly. Similarly, does it make any sense to select a time step of 0.01 s in the second case, when one revolution takes $10^{13}$ of that time? No, of course not, you would be throwing a baby with the water by taking such a huge time step.

So, the general rule is that **the time step should be small compared to the time scale**, but not unnecessarily too small. In the case of the Moon orbiting the Earth, I would take a time step of, say, 1/1000th of the month (i.e. 43 minutes) or 1/10000th of the month (i.e. 4.3 minutes), but not smaller than that. Unless, of course, you would like to spend the rest of your weekend waiting for your program to complete just one revolution :-) 

The first question you should think about before selecting a time step is this: What is a characteristic time scale for a given physical problem? Estimate it the beforehand, perhaps very roughly. Once you have an idea of the time scale, only then choose your time step wisely.

Note that changing the time step in your program and changing the argument of the **rate(.)** command in VPython do NOT produce the same result. The rate affects only how fast your graphical objects move in the virtual world and how fast a computer completes your simulation. If you run the same code with different values of the rate, you will always get exactly the same numerical result at the end. On the other hand, changing the time step may influence physical results and produce undesirable effects: if your time step is too large, the result of the simulation may be completely unphysical. So, it is a good idea to select a small time step as described above (but not unnecessarily too small), and only then to adjust the argument of the **rate(.)** command to optimize your graphics output or to maximize the computer speed.