Subject: Re: Delayed plotting in object graphics
Posted by Mark Hadfield on Wed, 13 Dec 2000 01:52:15 GMT
View Forum Message <> Reply to Message

"Ben Tupper" <pemaquidriver@tidewater.net> wrote in message
news:3A36BBD9.1A4F589C@tidewater.net...
> Hello,
>
> I am converting some 2d direct graphics to object graphics.
> Past users of the program producing the graphics have wanted
> to plot the points in 'slow motion'. That was easy to brute
> force a lag using PLOTS and WAIT in a loop. Is there a way
> of doing this with object graphics?

Hmmm. Interesting. Some ideas:

1. Naive and ugly: Add the points to the scene one at a time; each time a point is added, redraw the scene and then wait. But once the scene has been constructed, every redraw will be instantaneous (well, as instantaneous as OG ever is).

2. Sophisticated and ugly: Add the points to the scene all at once, but with each point in a separate model. Make these models instances of a purpose-built subclass of IDLgrModel, say SlowModel. In SlowModel's class structure add a tag called delay, to be set to a sensible value in the Init method. Define SlowModel draw as:

```
pro SlowModel::Draw, oSrcDest, oView
   self->ILgrModel::Draw, oSrcDest, oView
   wait, self.delay
end
```

Then every redraw will be slowed down. (But you might be surprised how often OG scenes get redrawn, especially when printing them!)

3. Best I can think of: Achieve a similar effect by manipulating colour. Put all the points in the same atom (much more efficient!) and set the atom's COLOR property so that one point is in a conspicuous colour (red, black) and the others in a less conspicuous one (grey). Each time you want to highlight a different point, get, rotate and reset the COLOR property vector. Calling WAIT in a loop is a pretty ugly way of driving a process like this--with a bit of extra effort you can get a better result using widget timer events.

In general I find animation of one sort or another a very effective tool for helping visualise data. I have been experimenting with different ways of animating direct & object graphics (mainly the latter). You might want to look at some of my attempts at
http://katipo.niwa.cri.nz/~hadfield/gust/software/idl/. In particular the

MGHgrAnimatorBase class implements basic animator functionality with user controls and the MGHgrAnimator class is a general-purpose object graphics animator application. But I never thought of trying the sort of animation you describe and I don't have anything that would implement it directly.

Mark Hadfield
m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield/
National Institute for Water and Atmospheric Research
PO Box 14-901, Wellington, New Zealand