Mathematics 308—April 5, 1996

Drawing paths in 3D

I have added another routine to to the course library to draw paths in two dimensions, and several which you can use to draw various kinds of paths in three dimensions. The most useful one for 3D work was somewhat tricky to write. It does not rewrite the code for the two dimensional version of `mkpath`, but instead manages to incorporate it. This involves using a variable which is the name of the name of a routine, a second level of trickiness up from `mkpath` itself, which only uses the name of a routine. You can look at the PostScript source files to see exactly what they do, but this is not necessary in order to use them.

For two dimensions I have added a routine `mkpolypath` which has exactly the same calling set up as `mkpath`, but which ignores the velocity vector and just connects points by straight lines instead of Bezier curves. This to help you debug your program in a simpler context, and most likely to tell when your formulas for derivatives are incorrect — which for me is surprisingly often.

For three dimensions I have added routines

- `mkpath3d` has the almost the same calling set up as `mkpath`, except that (1) it is prefaced by a 'location matrix' which is as elsewhere an array of the form `[m t]`, where `m` is a 3×3 matrix and `t` an array of three numbers, and (2) the routine referred to in the call returns a pair of three dimensional instead of two dimensional vectors. The matrix m of the location matrix is the rotation component and t the translation component of a $3D$ affine transformation. The form of its use is thus

  ```
  location-matrix [parameters] parametrization-routine t0 t1 N mkpath3d
  ```

- `mkpolypath3d` is the $3D$ version of `mkpolypath`, which I mentioned above, with the same calling set up as `mkpath3d`.

- `mkpolygon3d` has no analogue in $2D$, where it is not necessary. It has two call parameters, a location matrix and an array of $3D$ points. It then draws the polygon described by these points after the location matrix has been applied. This sort of thing is not needed in two dimensiosn because of the built in operators `lineto` etc. together with the built in graphics context.

  ```
  location-matrix [[x y z] ...  [...]]  mkpolygon3d
  ```

In addition, there are two routines which you can use to switch between projection and perspective. To do this you just put in lines `use-perspective` or `use-projection`. *You must put in one or the other!*

The two dimensional routine is in a new version of the file `mkpath.inc` and the rest are in a new file `mkpath3d.inc`. For many of you these routines will make your projects considerably easier to do. For me they will, I hope, make it easier to read them.