About the Surface Category

A surface is described by a map of a rectangle in the space R^2 with coordinates (u, v) into the space R^3 with coordinates (x, y, z). The rectangle is the product of the intervals $[uMin, uMax]$ and $[vMin, vMax]$, and each point (u, v) in the rectangle is mapped to a 3-dimensional point P with coordinates $(x(u, v), y(u, v), z(u, v))$. The mapping is given by writing Pascal procedures that define the components of P as functions of u, v and nine global "parameters", called $aa, bb, cc, dd, ee, ff, gg, hh,$ and ii. Most surfaces need only 1,2,3 or even none of these parameters. (For example, an ellipsoid depends on three semi-axes, aa, bb, cc).

To discretize the surface, the domain in (u,v) space is divided into a grid which has $uResolution$ equi-spaced points in the u-direction, running from $uMin$ to $uMax$, and $vResolution$ equi-spaced points in the v-direction running from $vMin$ to $vMax$. This grid divides the (u, v) domain rectangle into subrectangles. When a surface is "Create"-ed, the mapping $(u, v) \mapsto (x, y, z)$ is applied to each of the grid points, so each of the subrectangles in (u, v)-space is mapped

to a quadrilateral in (x, y, z)-space called a "patch". The collection of these patches is what actually represents the surface immersed in 3-dimensional space. (For more details on the discretization process, see the discussion of "Resolution Factor" in the "View" Help panel).

To "see" the surface, it is necessary to make a 2-dimensional projection of these patches on the screen. To do this, the program projects each patch (either in perspective or orthographically) onto a plane ("ImagePlane") at a distance ("FocalLength") from the "ViewPoint". ImagePlane is orthogonal to a direction called "ViewDirection", so the line from ViewPoint in the direction of ViewDirection meets the ImagePlane at a point called "ImagePlaneCenter" at a distance FocalLength from ViewPoint. ImagePlaneCenter is always at the center of the Graphics Window. Of course, a perspective projection has its center of projection at ViewPoint. There is a distance, called ClipDistance, usually set to a fraction (ClipRatio) of FocalLength, and a patch is "clipped" (i.e., not projected) if any of its vertices lie behind the ClipPlane (the plane parallel to the Image Plane at a distance ClipDistance from the ViewPoint). One more

piece of data is needed to render a patch on the computer screen, namely the relation between distances in the ImagePlane, and distances on the monitor. This is defined by an integer called "Graphic-Scale", which gives the number of pixels per unit length on ImagePlane—i.e., on a "standard" monitor, with 72 dpi resolution, a line segment of unit length in ImagePlane will have length Graphic-Scale/72 inches on the monitor. (The "normal" value of Scale is chosen to be one-tenth the sum of the height and width of the Graphics Window in pixels. This has the effect of scaling objects proportionally to the window size, and usually gives a value of scale between 70 and 100.) We've cheated a little. We still need one final piece of information before we can actually render a surface on the screen, namely the direction on the ImagePlane that will point up on the monitor.

By default, patches are sorted by the distance of their centers to the ViewPoint, and are drawn on the screen in order of farthest to nearest ("Painter's Algorithm"). (The Painter's Algorithm occassionally has problems, it fails along a line of double-points of an immersed surface, but this failure is usually not serious at reasonably high resolutions).

By default, patches are illuminated by a white light coming from the ViewPoint, and three colored lights, one red, one green, and one blue coming from three different directions. These colored lights help give a realistic three-dimensional look to the surface. The brightness and color of each patch depends on the cosine of the angle the normal to the patch makes with the in-coming light directions.

For some purposes it may be better to make the patches "transparent" and view the surface as if it were made of chicken wire. To switch back and forth, choose either "Wire Frame" or "Patch" from the View menu.

When you have selected a particular surface from the Surface menu, a version with certain default parameter values will be displayed. You can then choose "About this Object" from the Action Menu to see how the surface depends on the parameters, after which you can change these parameters in the Settings Menu before re-creating the surface.

The program can also interpolate linearly between two surfaces of the same family that you can set by choosing "Set Morphing..." in the Settings Menu. The number of steps in the "morphing" is the Num-

ber of Frames in the filmstrip, an integer that you can also set. Playing back the filmstrip gives a "movie" of the surface changing gradually ("morphing") between the initial and final surface. It is also possible to rotate the surface in space, and form a filmstrip of this rotation that can be played back. This gives a very good illusion of seeing the three-dimensional nature of the surface., but is even easier to just rotate the surface by dragging the mouse across the screen in a manner that is easier to figure out by experiment than to explain. If you have red/green glasses you should do the animations in stereo vision for the most realistic effects. (Switch back and forth between stereo and mono vision using the View Menu.)

A user can define a surface by choosing one of the User Defined... items from the Surfaces menu. This will bring up a dialog that will permit one to create algebraic expressions involving the two surface variables u and v and the nine parameters, aa,bb, ... , ii. There is some checking done to see that these are well-formed formulas, but you should be careful to avoid other errors (such as an implicit division by zero) that could have unpredictable consequences. If you mess up badly in editing the expressions, press

the Default button to reset the expressions to their original values. When you have finished editing, either press the Create button to immediately create the surface, or press the OK button and make various choices from the Settings menu before choosing Create from the Surfaces menu. Note that if you want to create the graph of a function, $f(x, y)$, i.e., display the surface $z = f(x, y)$, then you can use the parametric equations $P.x = u$, $P.y = v$, and $P.z = f(u, v)$;

The User Defined... surface feature uses David Eck's expression parsing and evaluation routines. Expressions can include the operators: $+$, $-$, $*$, $/$, and $\uparrow$, and the available built-in functions are: sin, cos, tan, cot, csc, sec, sinh, cosh, tanh, coth, arcsin, arccos, arctan, arccot, exp, ln, sqrt, cubert, abs, round, trunc, and the three basic Jacobi elliptic functions, sn, cn, and dn.

After a parametric surface is created, the Show Parallel Surfaces menu item of the Action menu will become enabled. Choosing it will first show the two focal sets of the surface and then start an animation showing the parallel surfaces to the given surfaces (i.e., the surfaces at a constant distance out along the surface normal). If you hold down the Shift key

during the animation you will be able to see the focal sets until you release it (the parallel surfaces develop singularities where they meet the focal sets). Each press of the right-arrow key will speed up the animation, while each press of the left-arrow key will slow it down. Pressing the up-arrow key reverses the direction of the animation.