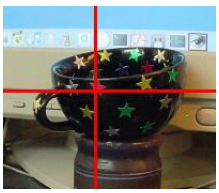
COMMERCIAL SCANNERS. There are scanners on the market which can scan 60'000 points per second. The model on the photo to the right costs currently 410,000 dollars (www.cyberware.com).

THE PROBLEM. Given a surface in space. Take a cup for example. How do we model the surface mathematically? Taking pictures of the cup only gives us two dimensional representations. Reconstructing the surface from two dimensional projections is called **3D scanning**.



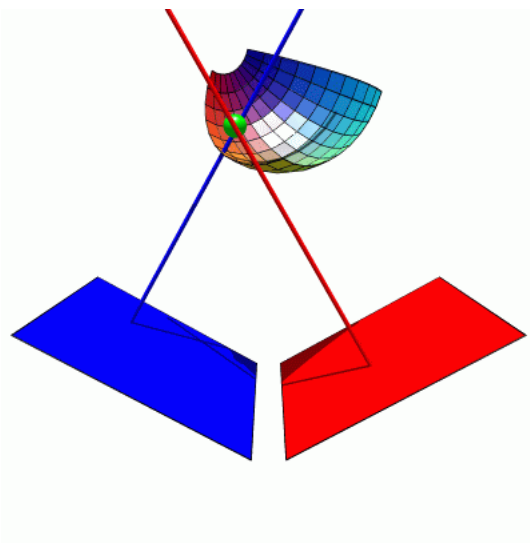
SOLUTION. One method is to make two or more projections. Each point in the photo determines a line in space. If we have at least two photos which contain both points, the point is fixed: the intersection of these lines is the point we are looking for. The next problem is then construct a surface from these points.

To see how this works in detail, we split up the problem into three subproblems.

PROBLEM 1) Given the camera position $C = (a, b, c)$ and the center $P = (x, y, z)$ of the object as well as a point $p = (u, v)$ on the film: find a line line $r \mapsto C + rU$ which passes through the point P, where $U = C - O$.

SOLUTION. If $U = C - O = (u_1, u_2, 0)$ and $N = (0, 0, 1)$. A unit vector orthogonal to u and N is $V = U \times N = (-u_2, u_1, 0)/\sqrt{u_1^2 + u_2^2}$. Then $P = C + Vu + Nv$.

PROBLEM 2) Given the location $C_1 = (x_1, y_1, z_1)$, $C_2 = (x_2, y_2, z_2)$ of the two cameras and the location $O = (0, 0, 0)$ of the center of the object. Given a point (u_1, v_1) on the first film and a point (u_2, v_2) on the second film. Find the point P on the surface.



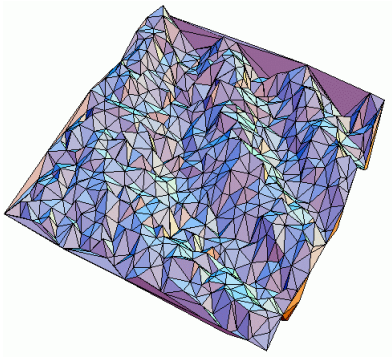
SOLUTION. $p = (u, v)$ on the first film determines by PROBLEM 1) a line $P + rU$. Similarly, a point q on the second film determins a line $Q + sV$ in space. Theoretically, these two lines meet in a point $X = (x, y, z)$. However, due to errors, these two lines will miss in general. We assign the best possible choice, the point closest to both points. Because we know $u_3 = u_1 \times u_2$ the direction of the closest connection between the two lines, we have to solve $C_1 + su_1 + tu_3 = C_2 + ru_2$ to get the point $X = C_1 + su_1 + (t/2)u_3$. This is a system of three equations for the three unknowns t_1, t_2, t_3. We can solve this system (in Lagrange multiplier problems, we solved more complicated nonlinear systems of equations).

FAMILIAR EXAMPLE. If the camera C_1 is on the x axes and takes a shot of the $y - z$ plane, then the point (x, y, z) will be the point (y, z) in the photo. Similarly, if C_2 is on the y axes, then a point (x, y, z) will be the point (x, z) on the photo. Ideally, if the point is from both sides, then (x, y, z) is determined.

PROBLEM 3) Given a set of points $P_i = (x_i, y_i, z_i)$ on a surface. Model the surface as a set of triangles.

SOLUTION. The solution to this problem is provided by an algorithm called **Delaunay triangularization**. It produces a desired set of triangles.

Fortunately, Mathematica has already built in a function "TriangularSurfacePlot" which is contained in the DiscreteMath'ComputationalGeometry' module. The figure to the right shows a triangularsation applied to a random surface.



STEP 1. DATA INPUT (C PROGRAM). A GUI xwindows C program 3dscan.c helps us to enter the data with a mouse. The two pictures are scaled, concatenated to a single picture and displayed. The program 3dscan.c allows to enter pairs of points with the mouse. The data are written onto a text file "result.dat".

STEP 2. DATAPROCESSING (MATHEMATICA). Given the object $O = (0,0,0)$, the camera positions $C_1 = (C_{1x}, C_{1y}, C_{1z})$,$C_2 = (C_{2x}, C_{2y}, C_{2z})$ and the points (u_1, v_1) on the first film and (u_2, v_2) on the second film. The formula for the point $P(x,y,z)$ are obtained from solving $A_1 + t_1 * C_1 + t_3 * N = A_2 + t_2 * C_2$, where $A_i = C_i + u_i(C_i - O) \times N/|C_i - O| + v_i N$.
A mathematica program "stereo3d.m" takes the data in "result.dat" and writes a new file "3d.dat" which contains the points P_i in space.

Also the triangularisation is done with Mathematica. The program is called "triangular.m". It takes as in input the file "3d.dat" and outputs a Povray file "surface.inc" which is essentially a set of triangles written in form, Povray can read.

STEP 3. RENDERING (POVRAY). Finally, with all the triangles known, we run the data through a ray tracing program "surface.pov". This Povray program takes the file "result.inc" as an input and produces a picture "surface.ppm".

POVRAY. Povray is a free raytracing program. It allows to render three dimensional objects. To the right, you see a simple Povray program. It simply shows a red sphere on a white background.

```
camera { up y right x location <0,1,-3> look_at <0,-1,3> }
light_source  { <0,300, -100>  colour rgb <1,1,1> }
#background { rgb <1,1,1> }

#declare r = texture {
  pigment { rgb <1,0,0> }
  finish { phong 1.0 ambient 0.5 diffuse 0.8}
}

sphere {<0,0,0>,1 texture{ r } }
                                          9,0-1         Top
```

THE LANGUAGE POVRAY.

$< a, b, c >$	point or vector (a,b,c)
sphere$\{P, r\}$	sphere centered of radius r centered at (a,b,c)
box$\{P, Q\}$	rectangular box with diagonal corners P,Q
cylinder$\{P, Q, r\}$	cylinder of radius r over segment P Q
plane$\{N, d\}$	plane with normal vector N, in distance d from origin
triangle { P,Q,R }	triangle with endpoints P,Q,R

One can take intersections, unions, differences of objects. Furthermore, every object can be scaled, rotated and translated.
In the example above, you see how to apply textures. Colors are represented as vectors $< r, g, b >$, where r is "red" and g is "green" and b is "blue". The addition of colors is "additive". For example $< 1, 1, 0 >$ is yellow. The best way to learn the language is to look at examples. There are thousands of examples available on the web. The official distribution of Povray includes many examples and contains a nice manual.