Anders Thorup: Input for 3D splines

The following pages contain examples of the use of a very simple Pascal program, `3dspline.pas`. The program is designed for generation of input data for curve drawings in TeX, and it requires (knowledge of) the TeX-package `spline.sty`. The routines in the Pascal program are centered around one procedure:

`cproj`.

To describe it, assume that the coordinates of a center in 3-space is given. Associate with it the central projection from the center onto the plane through the origin perpendicular to the center. Then the call `cproj(x)`, for a point `x` in 3-space, computes the image of `x` under the projection, and writes the two coordinates of the image point to an output file. The purpose is to use the procedure to generate coordinates that may serve as input for various spline commands and other curve drawing commands from the spline package. The output coordinates depend on the choice of center, and the following pages show how the output varies under changes of the center. The source code of the Pascal program contains several routines for manipulating with vectors in 3-space. The body part of the program produces several examples.

As mentioned above, the macros of the Pascal program are designed for the generation of input data for curve drawings in TeX; however, the body part of the program as it is generates a complete TeX file. In fact, all the pages you see are in the source code of the program, and this whole document may be reproduced as follows:

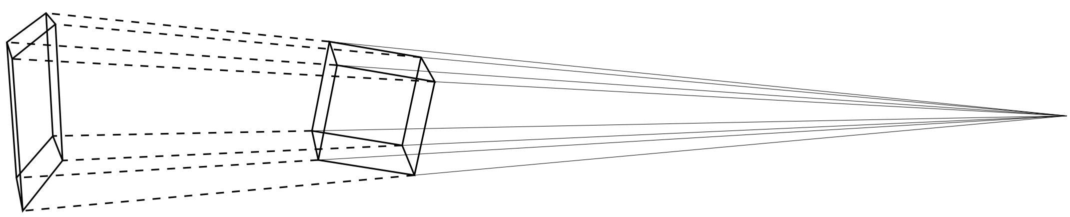
(1) Compile the Pascal program `3dspline.pas`

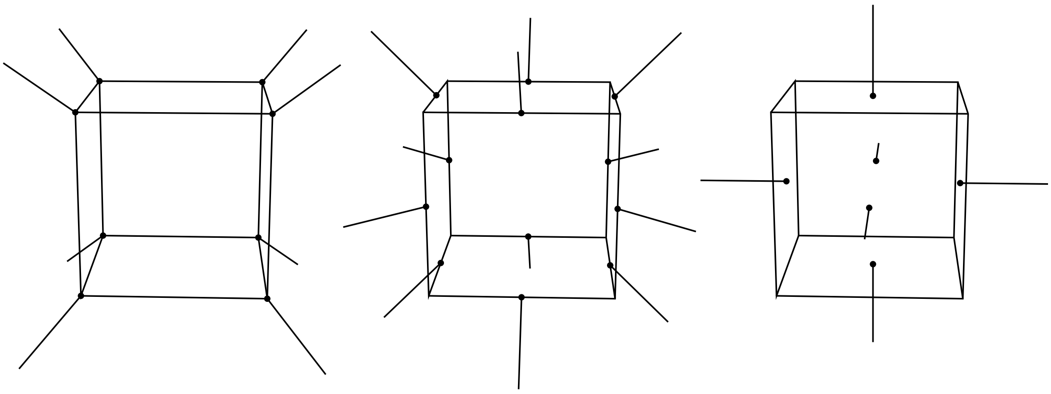
(2) Run the resulting executable; it will produce a tex file, `cproj.tex`, containing a lot of spline drawing commands.

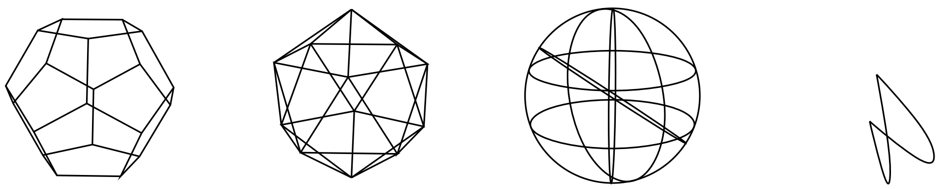
(3) Tex the file `cproj.tex` (with Plain TeX or AmSTeX); the file `spline.sty` is required.

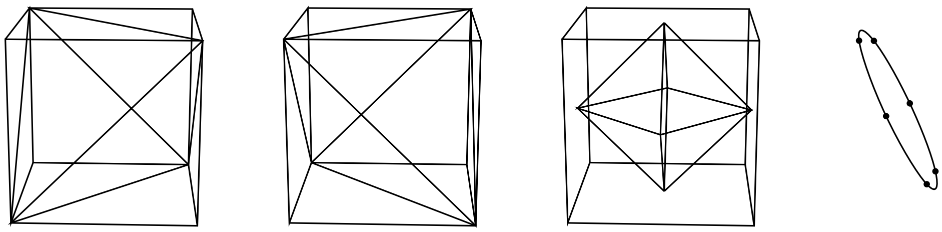
(4) View the result on your favourite device; as always when the spline package is used it is required that the interpretation accepts imbedded PostScript code as described in DVIPS.

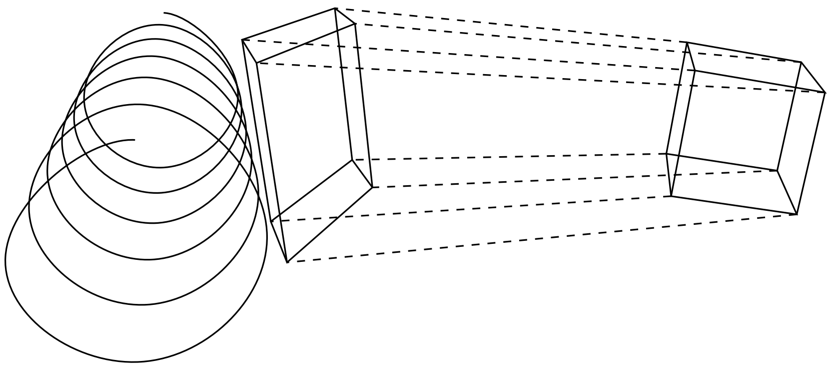
Anders Thorup
3 January 2005

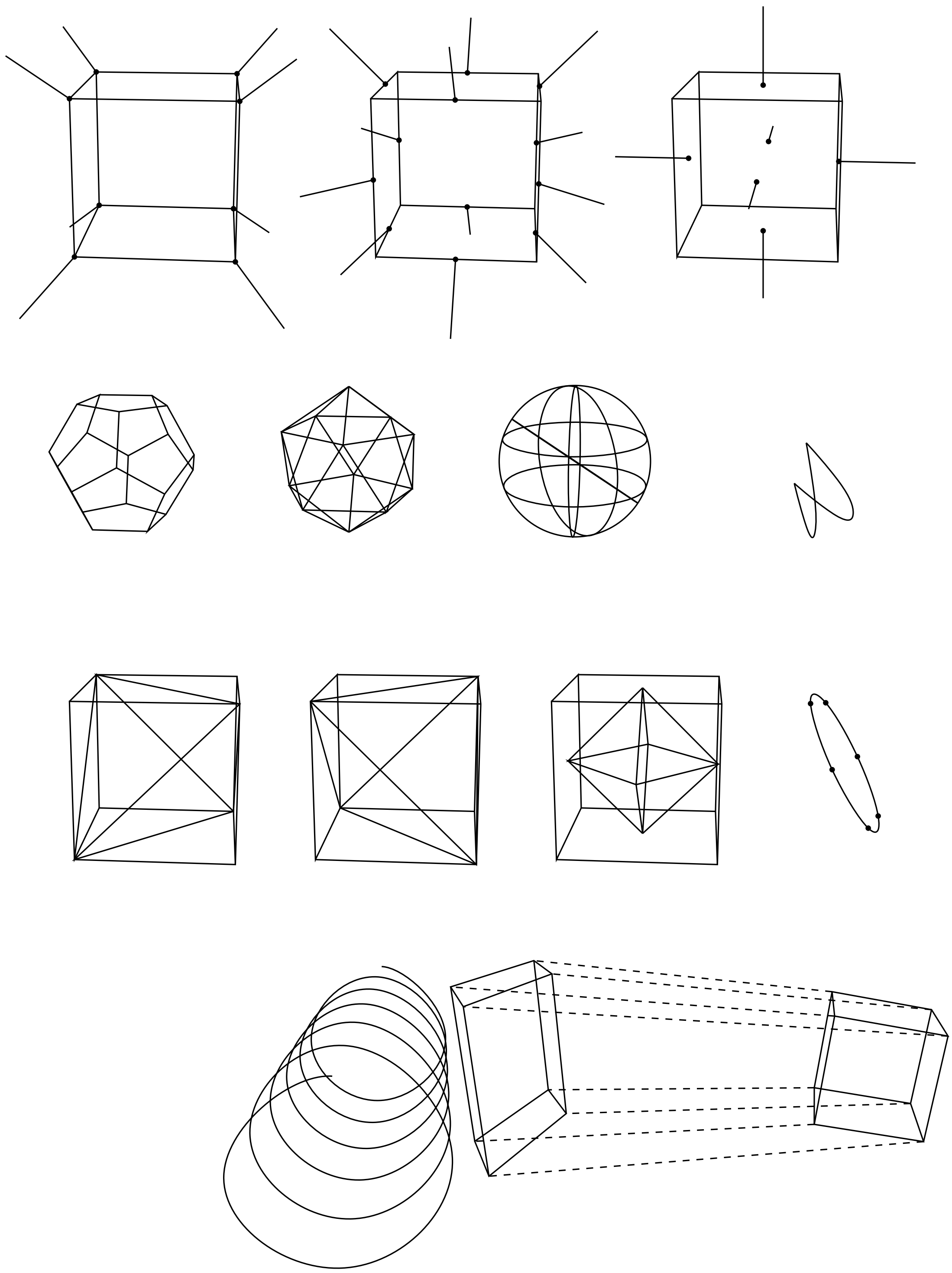


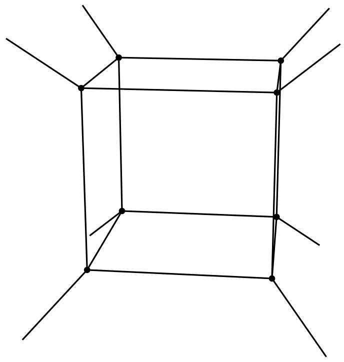
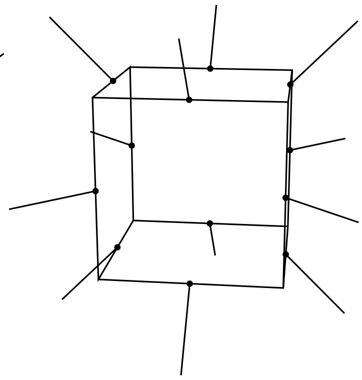
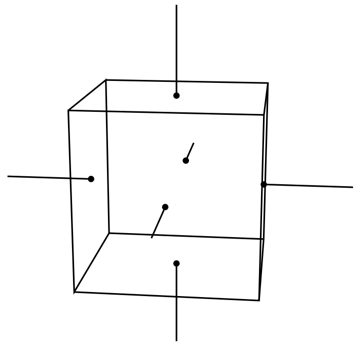
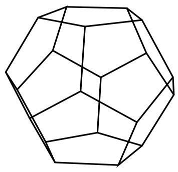
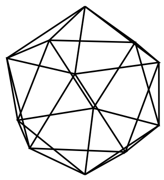
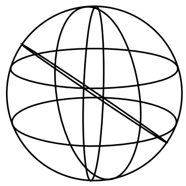
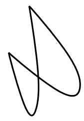
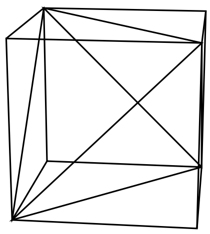
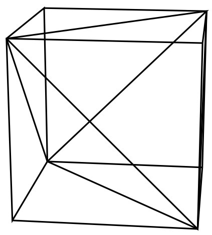
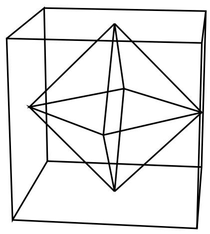
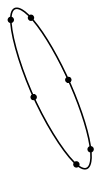
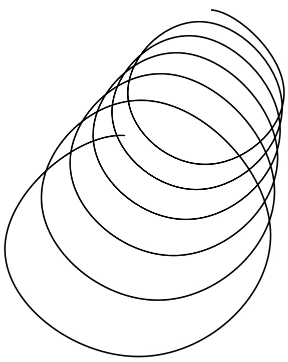
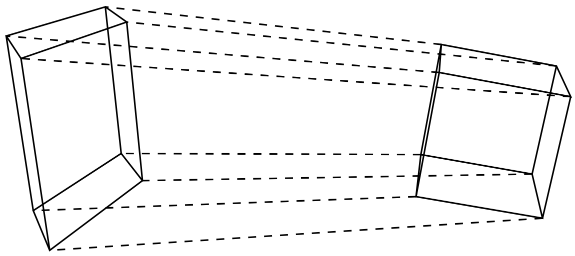


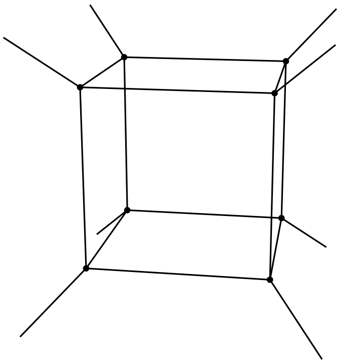
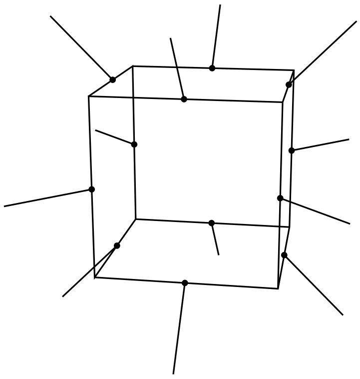
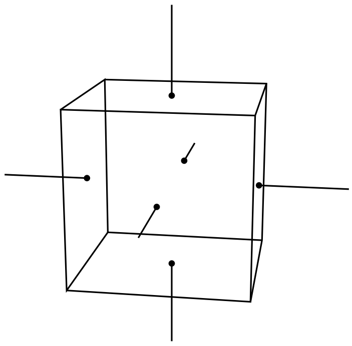


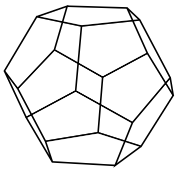
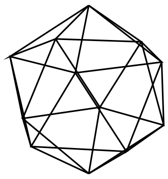
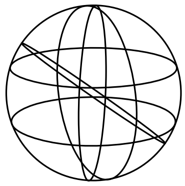
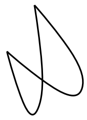


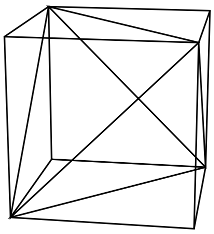
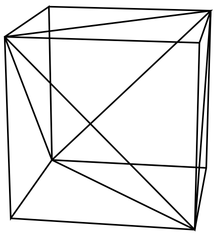
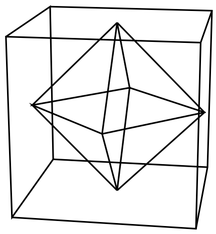
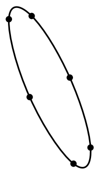


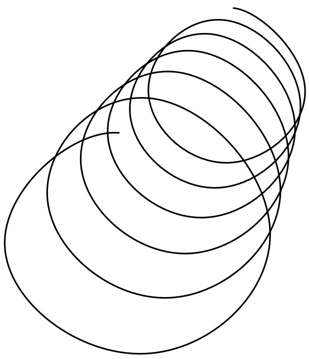
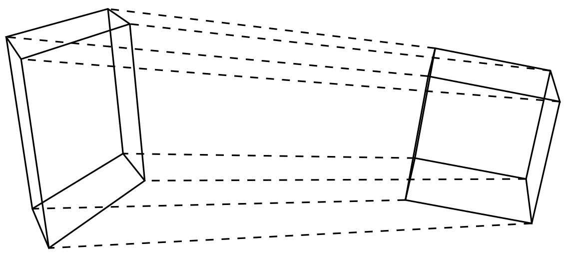


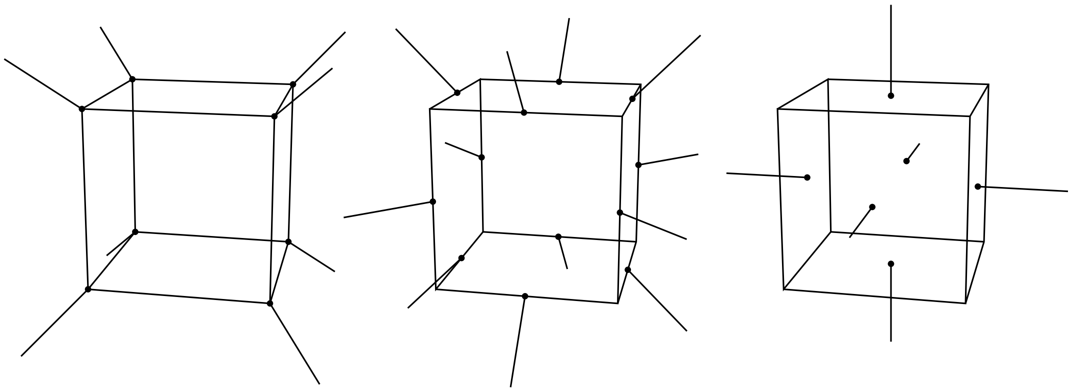














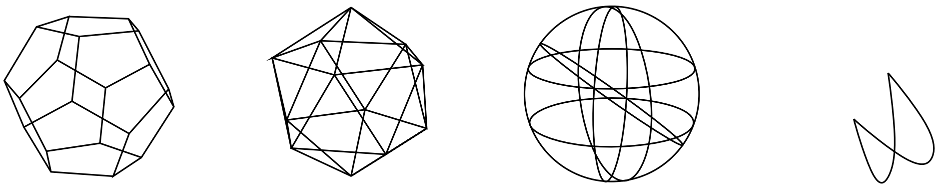




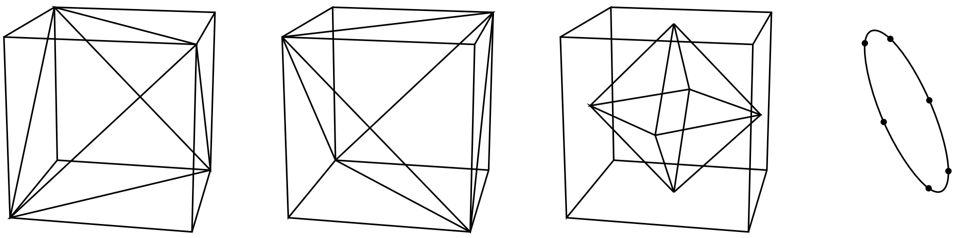





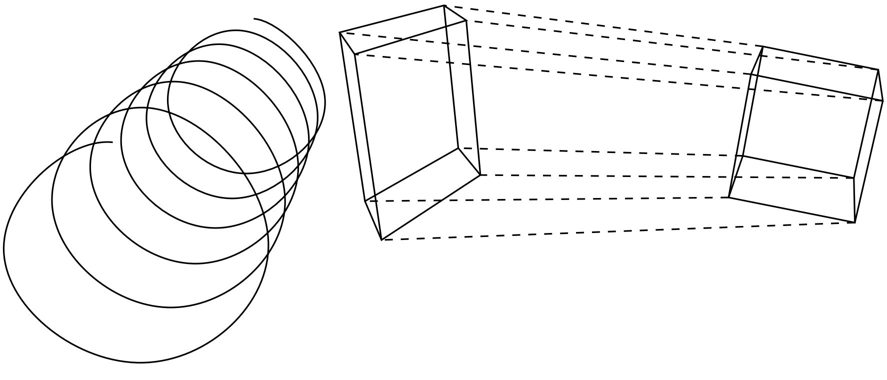





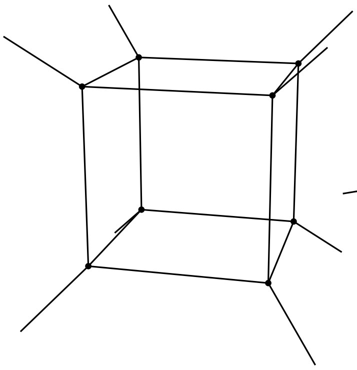
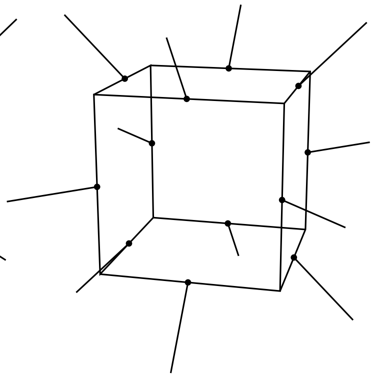
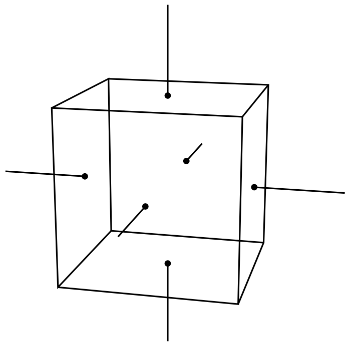



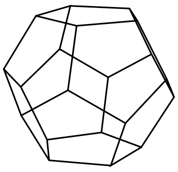
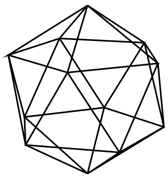
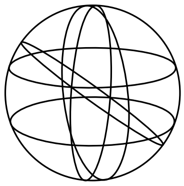
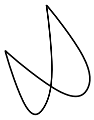


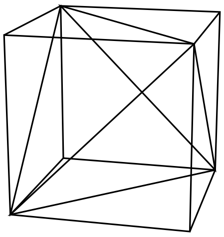
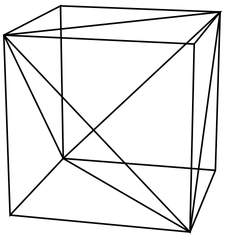
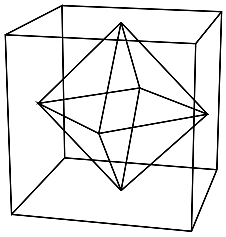
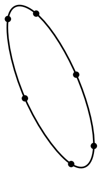


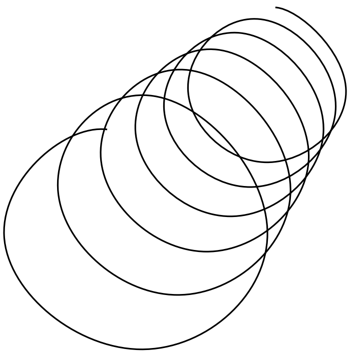
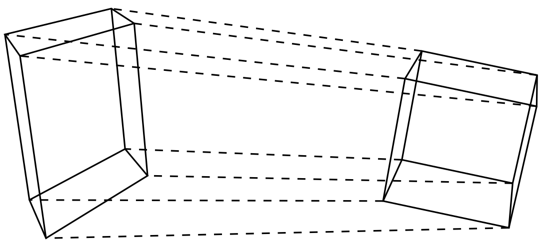


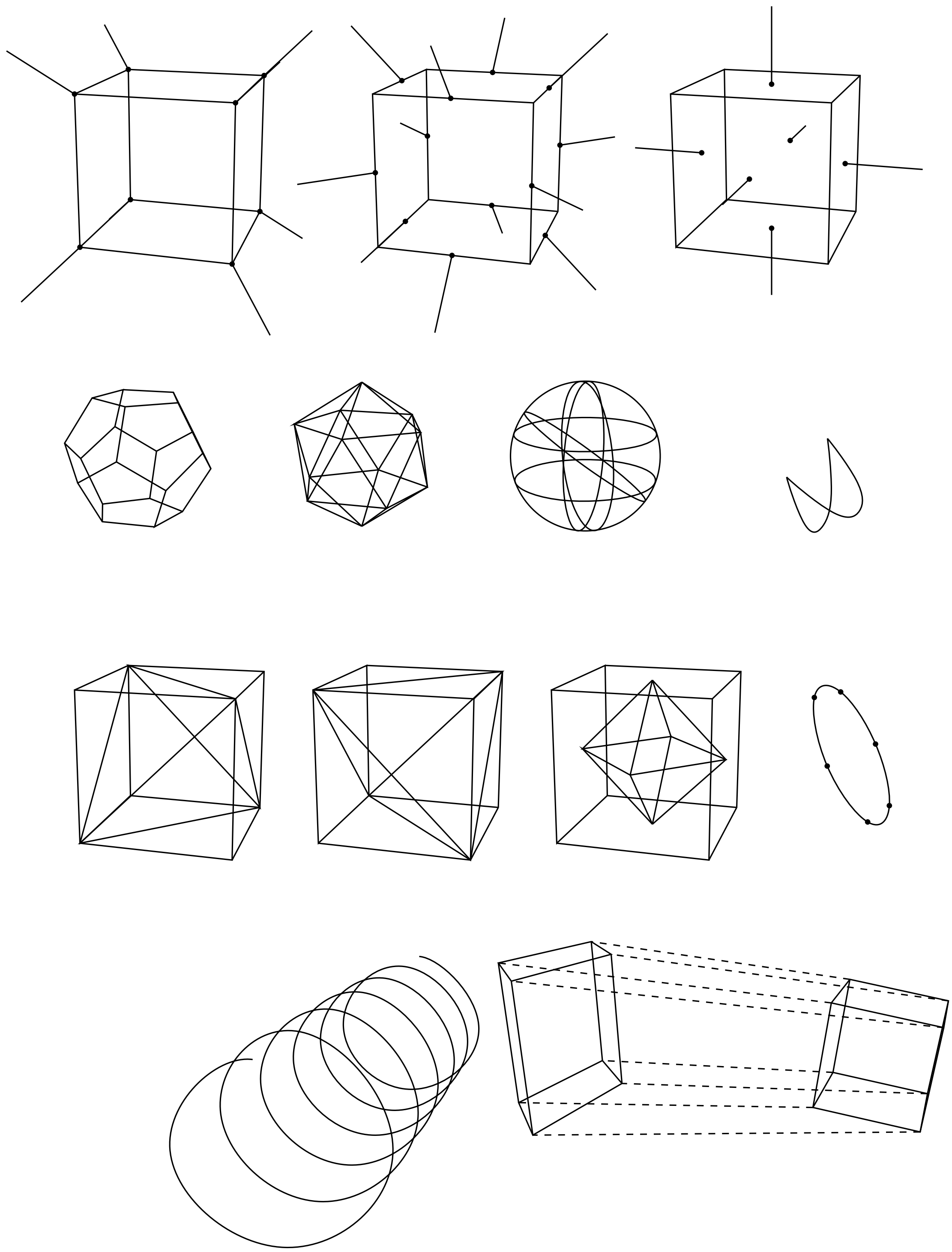


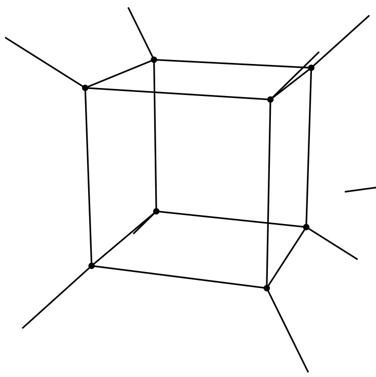
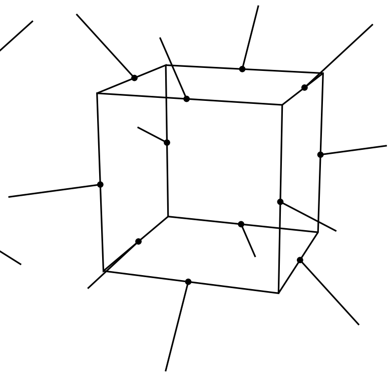
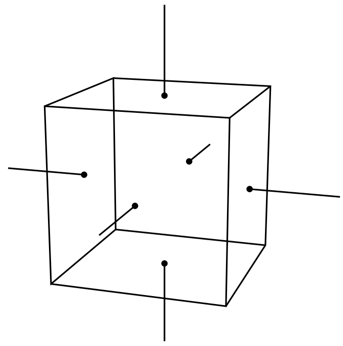




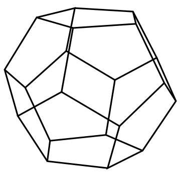
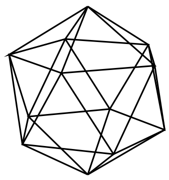
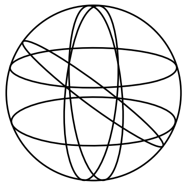
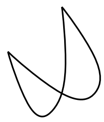





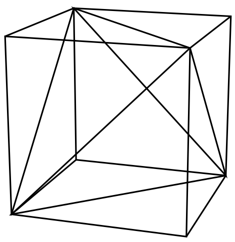
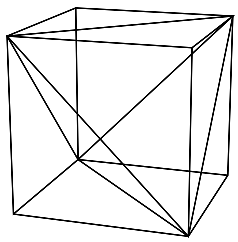
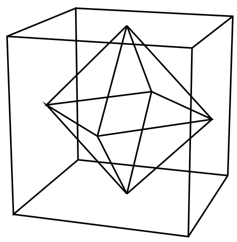
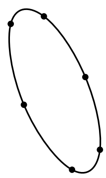





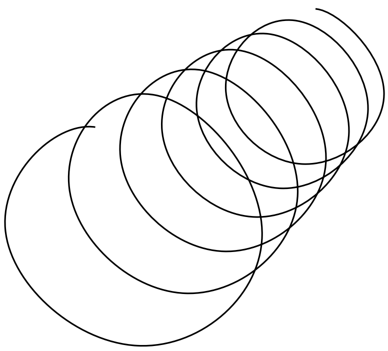
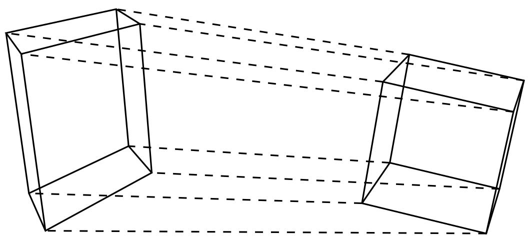



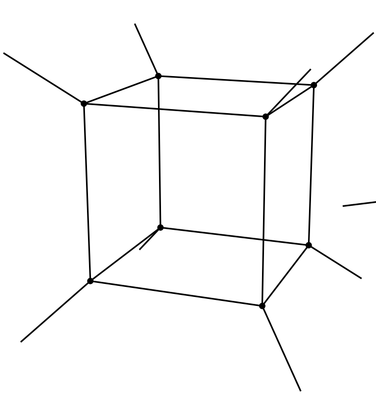
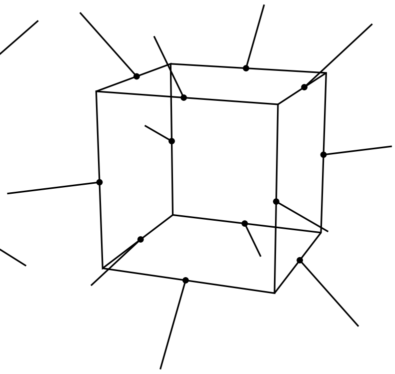
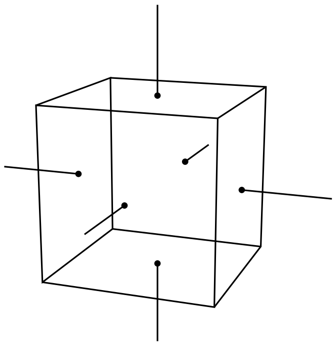


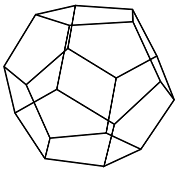
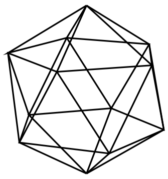
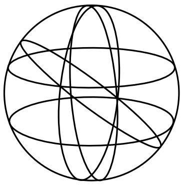
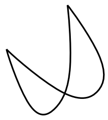




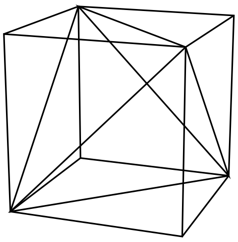
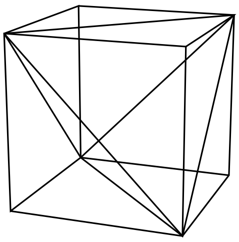
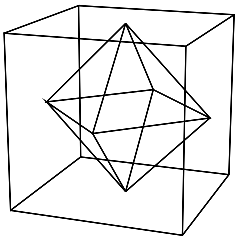
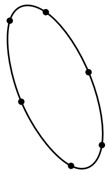





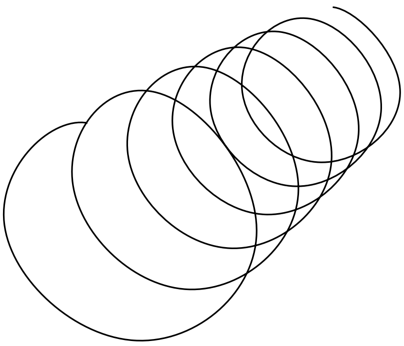
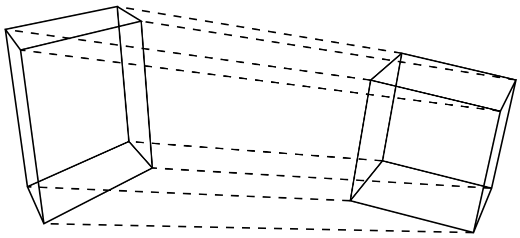





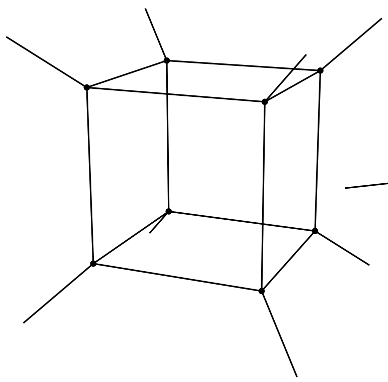
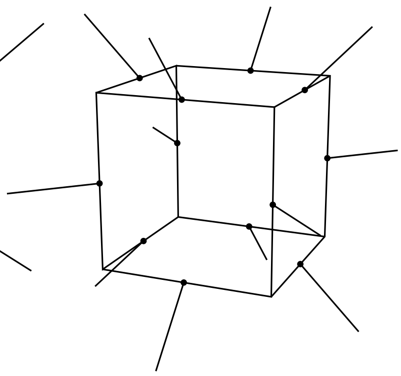
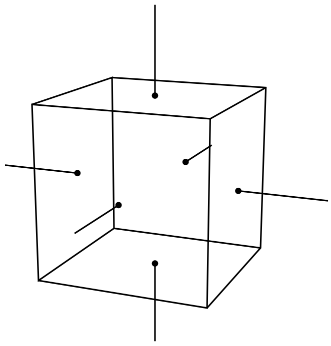



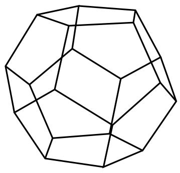
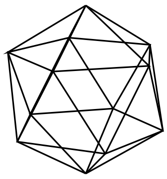
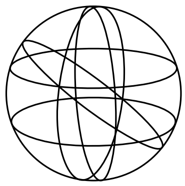




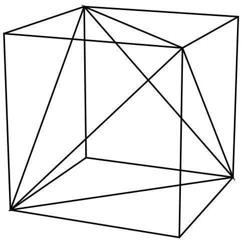
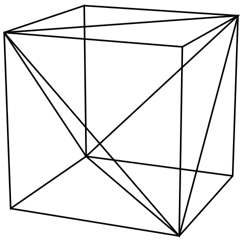
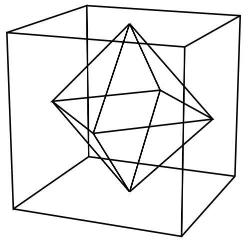
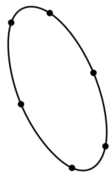





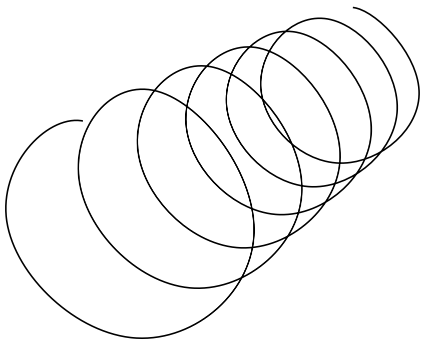
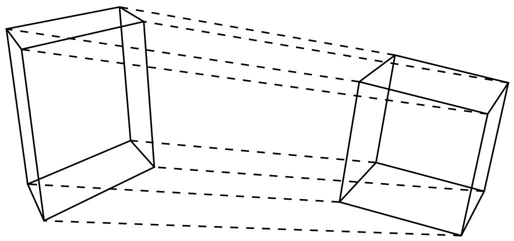





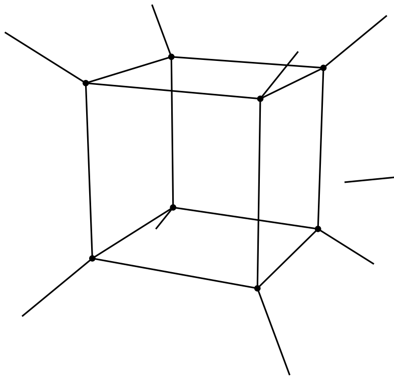
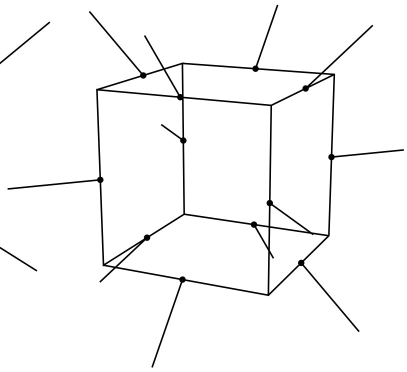
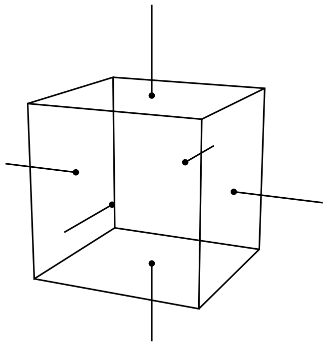



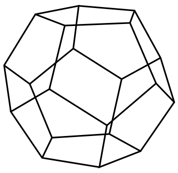
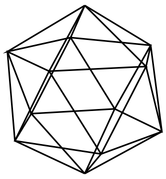
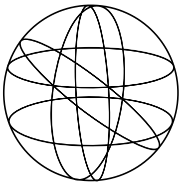




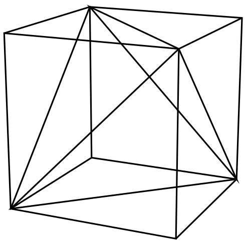
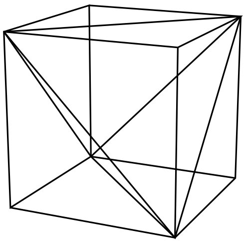
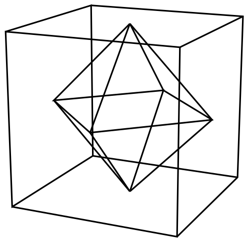
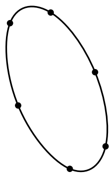




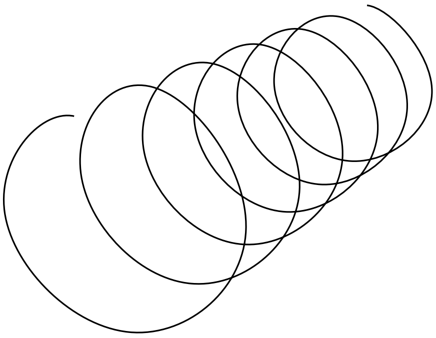
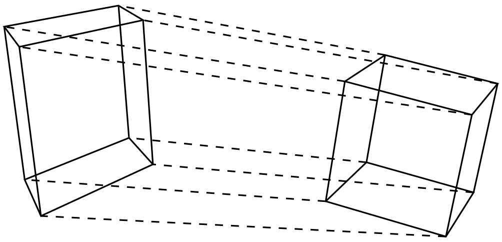






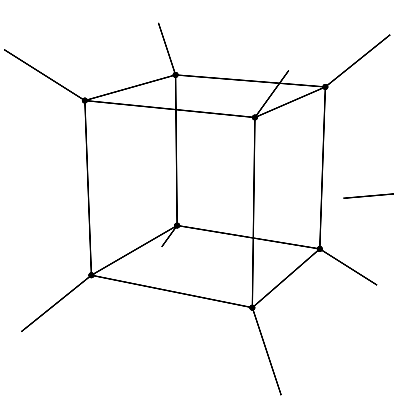



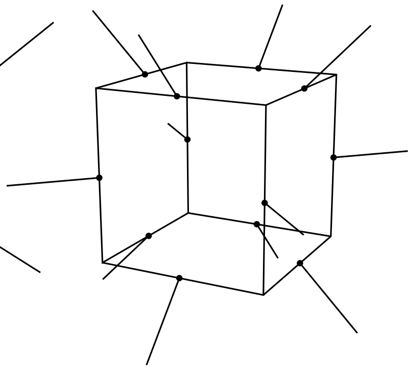




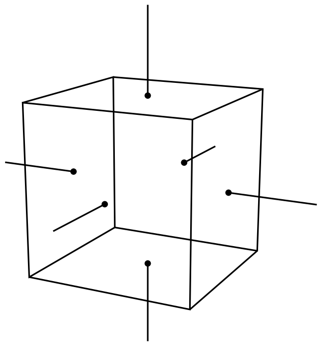




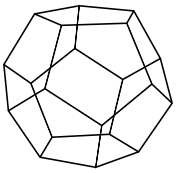






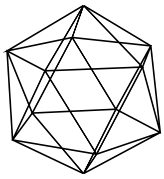



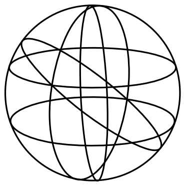


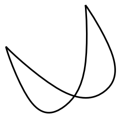


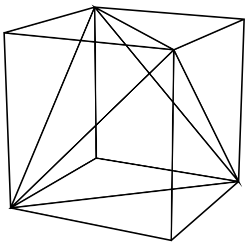


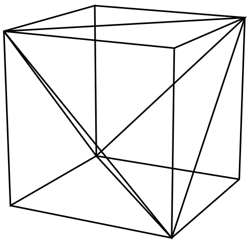


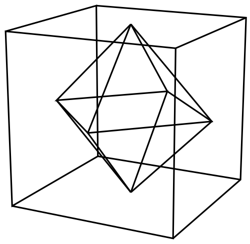


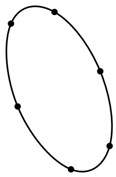


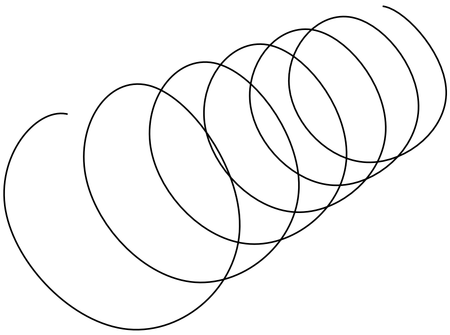


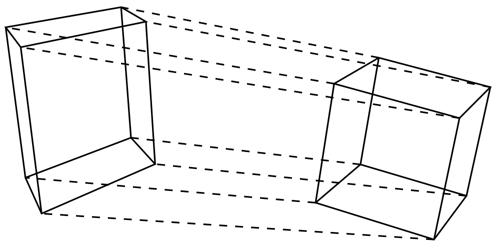


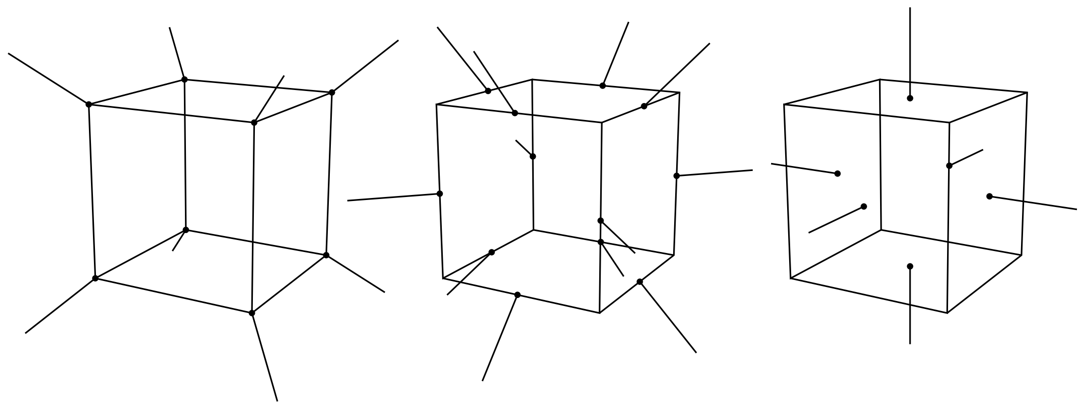


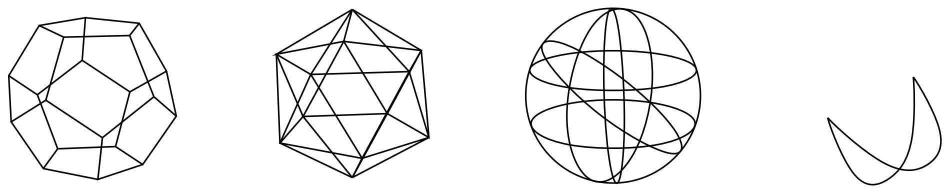


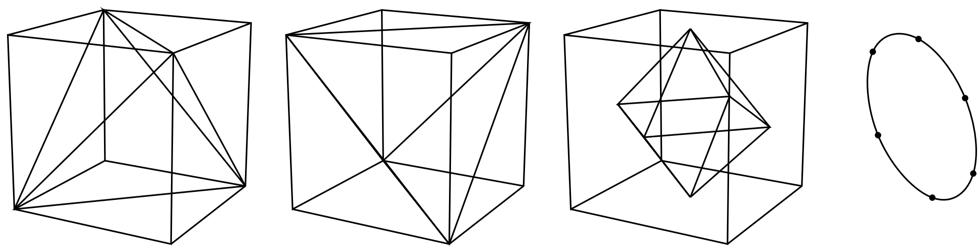


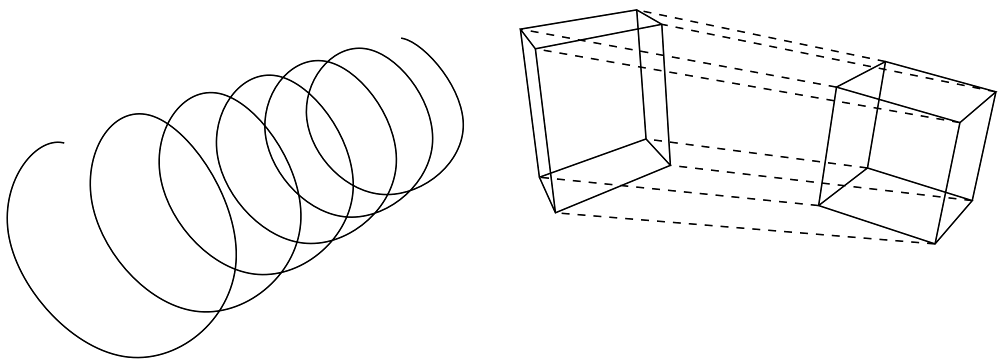


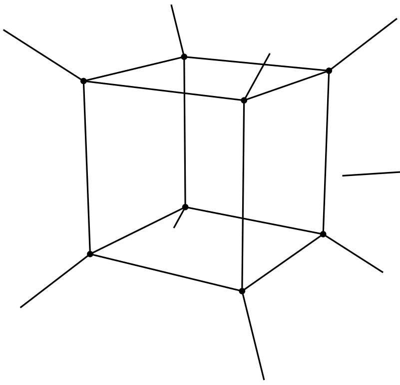


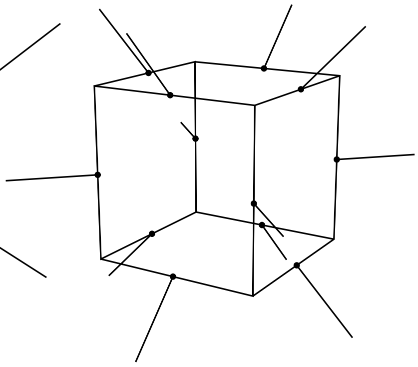


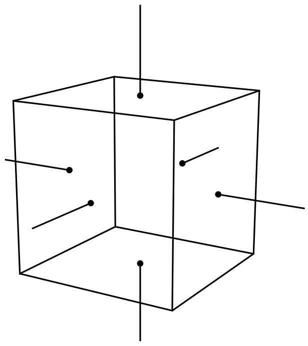


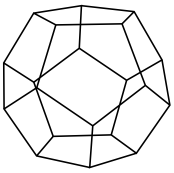


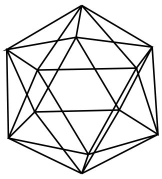


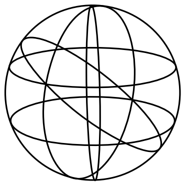


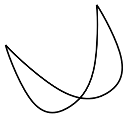


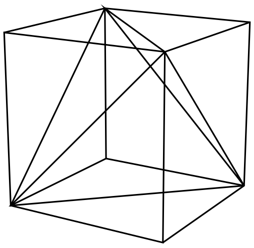


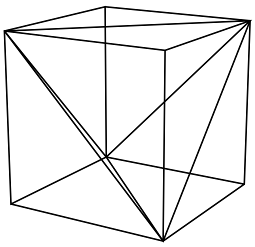


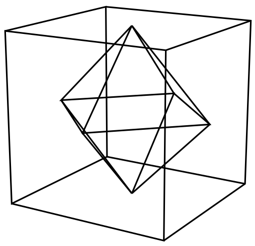


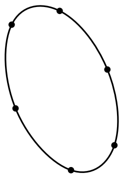


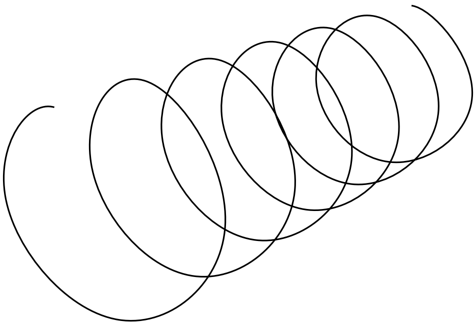


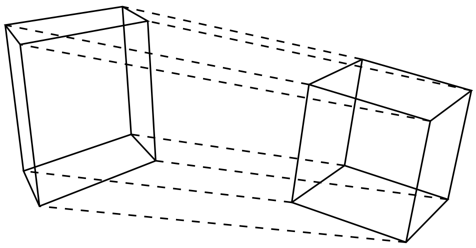


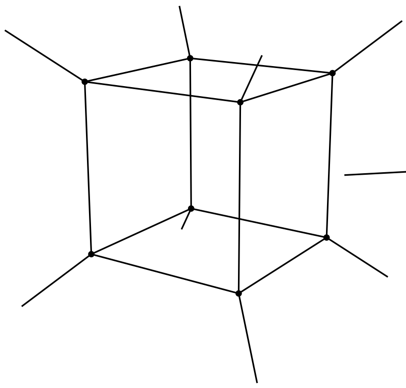
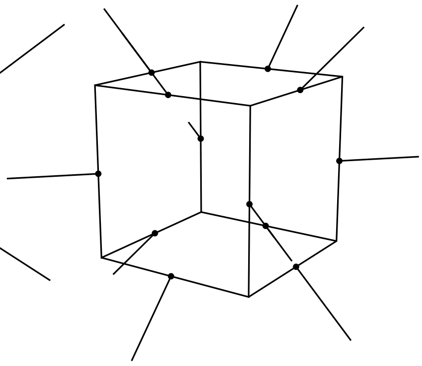
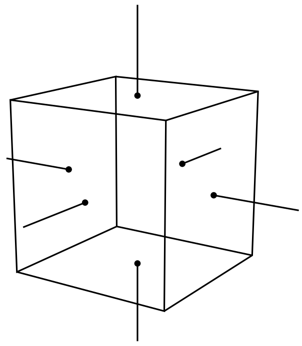


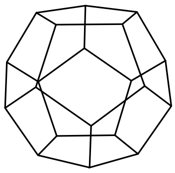
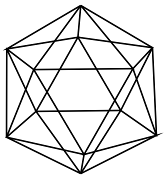
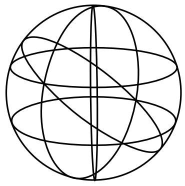
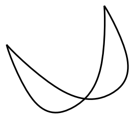


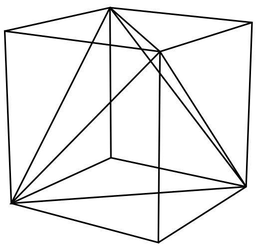
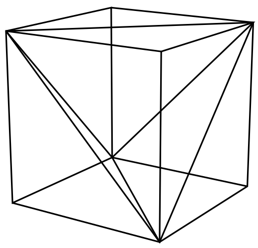
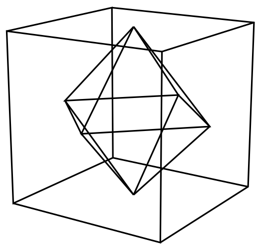
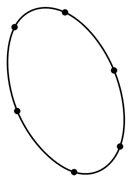


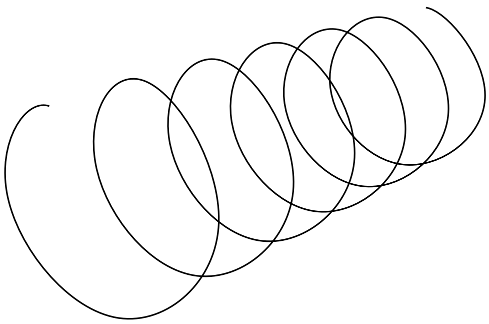
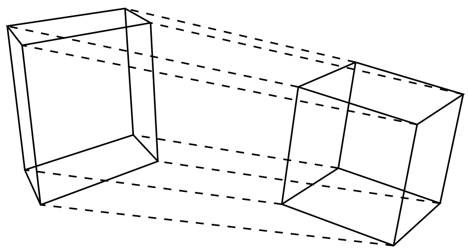


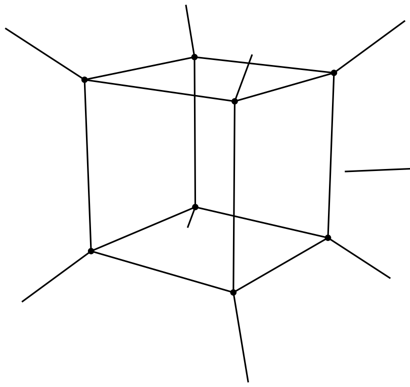
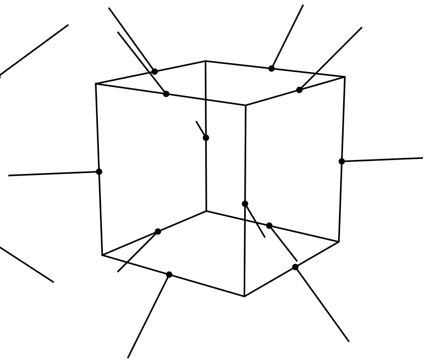
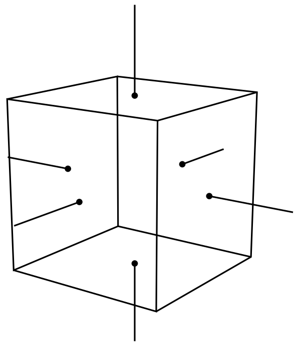


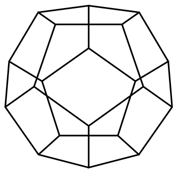
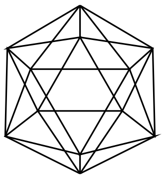
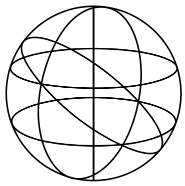
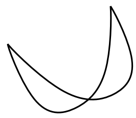




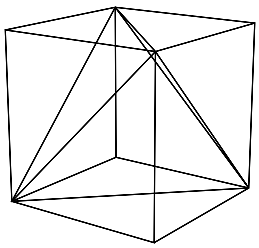
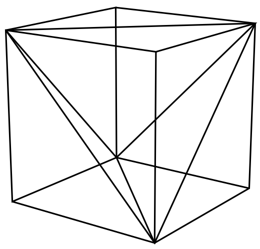
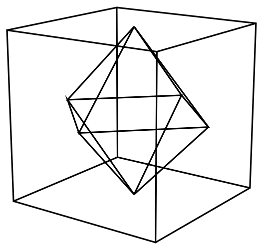
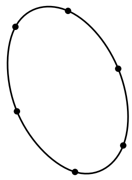





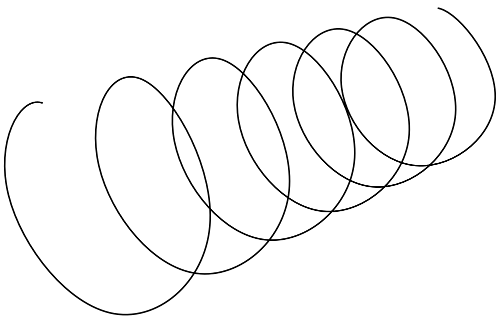
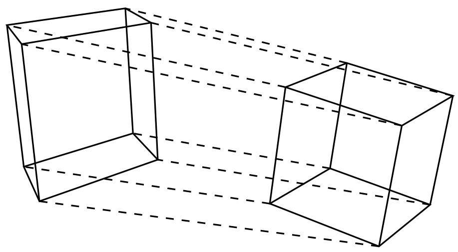





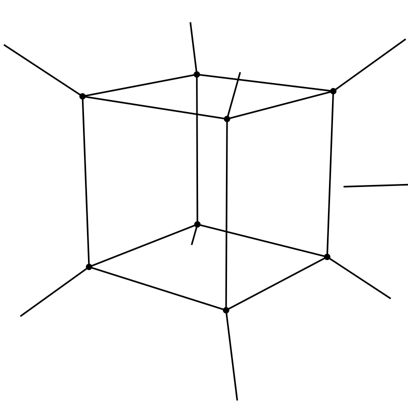
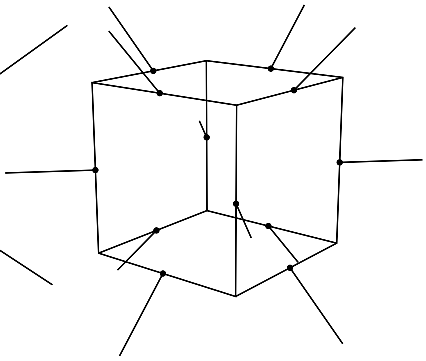
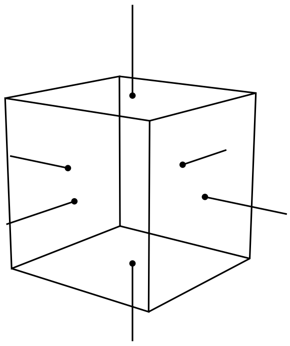



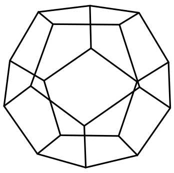
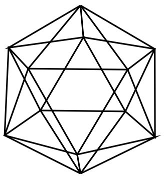
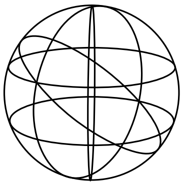
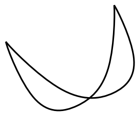




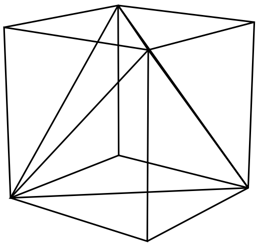
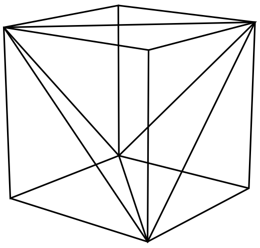
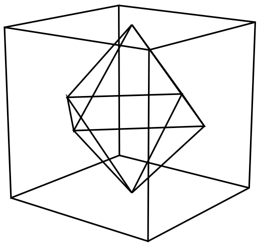
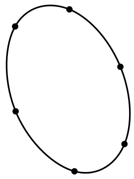





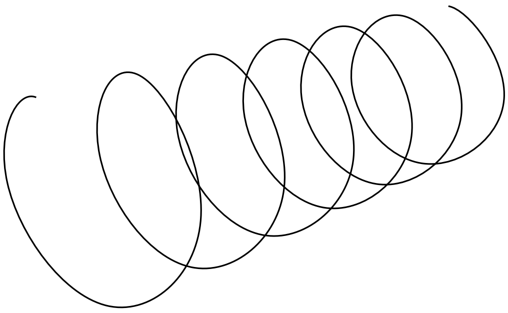
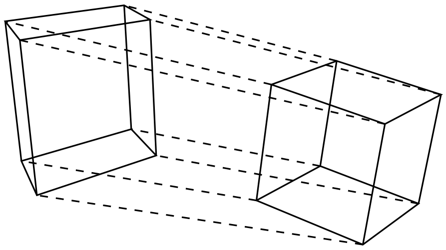





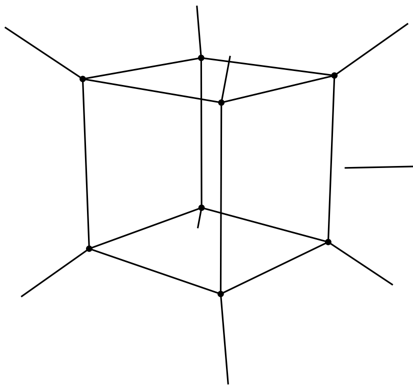
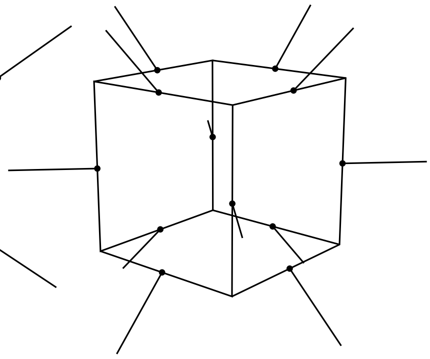
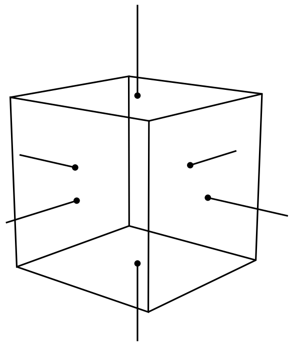



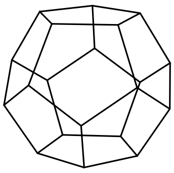
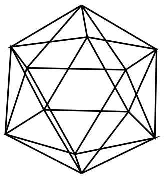
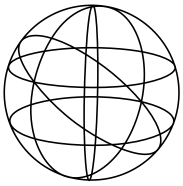
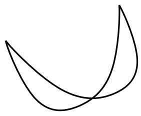




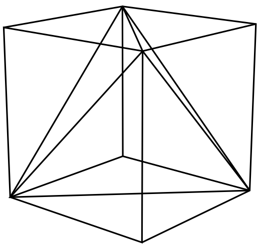
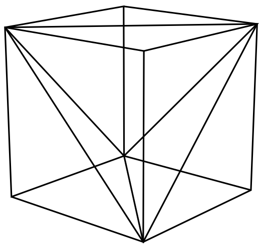
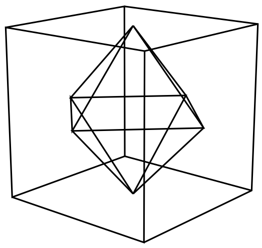
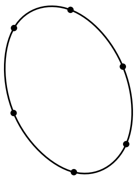





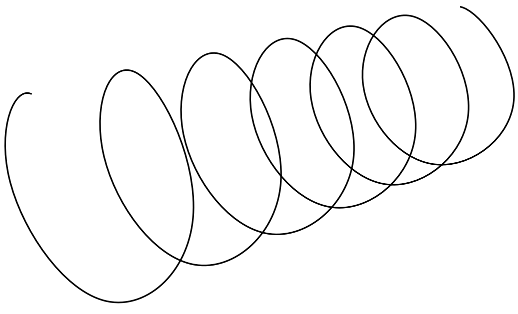
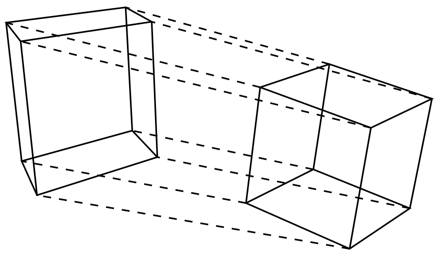





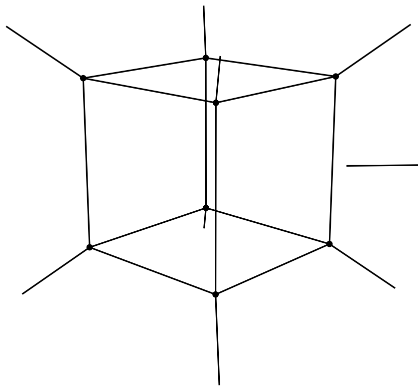
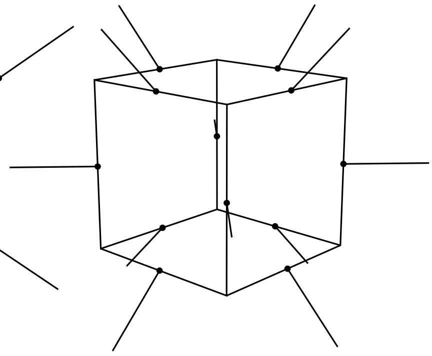
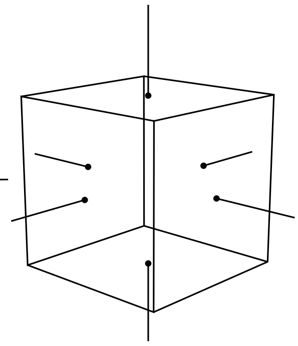



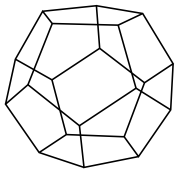
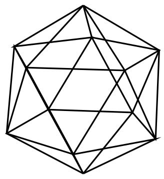
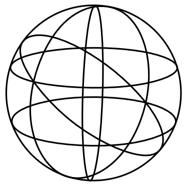
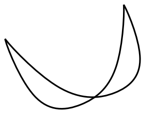




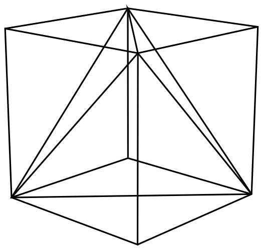
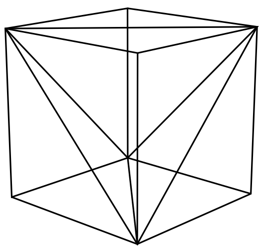
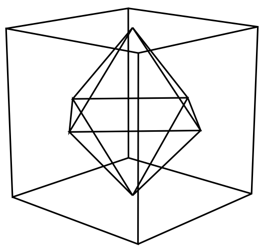
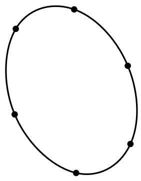





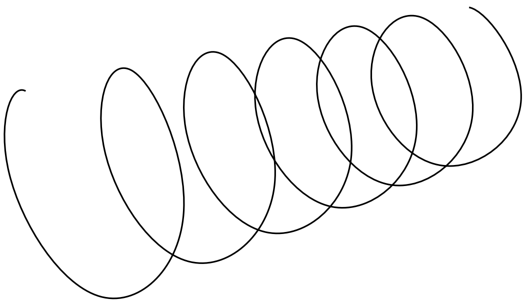
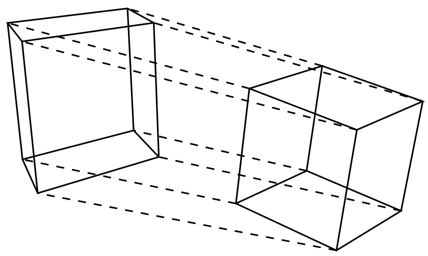





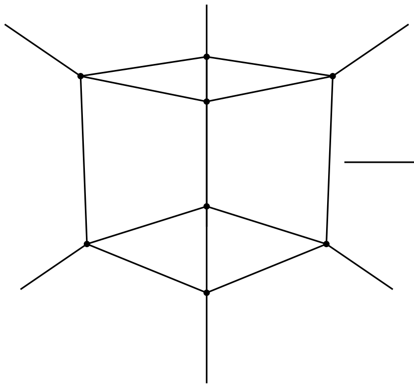
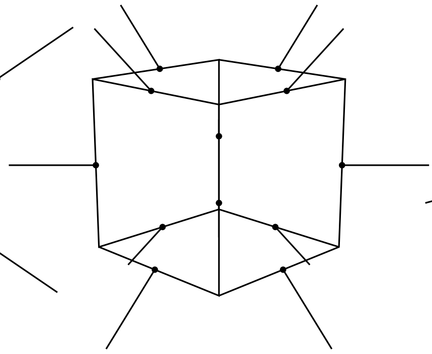
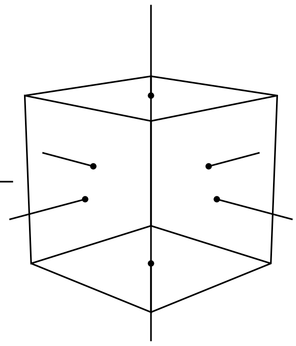



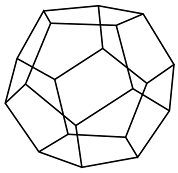
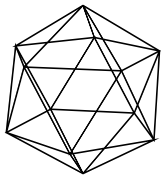
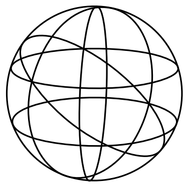
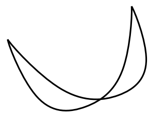




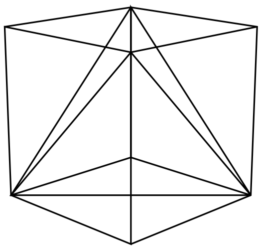
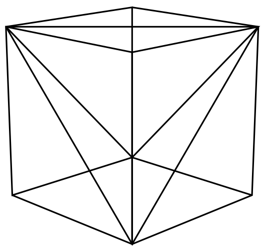
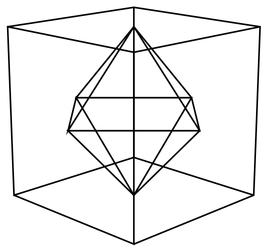
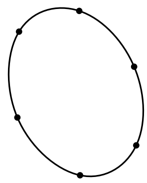





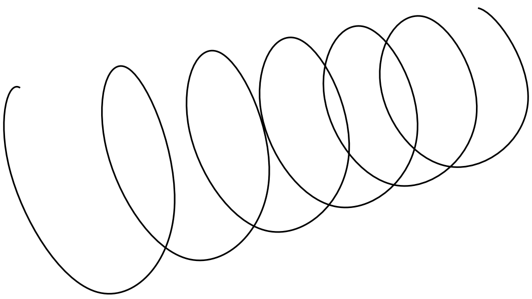
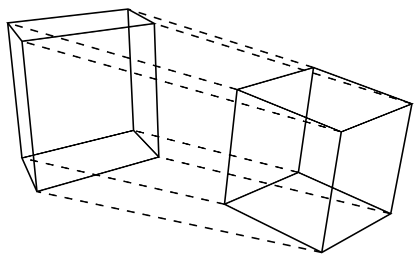





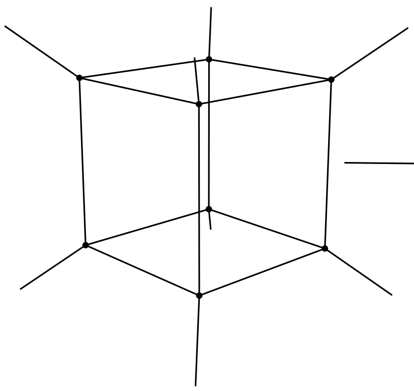
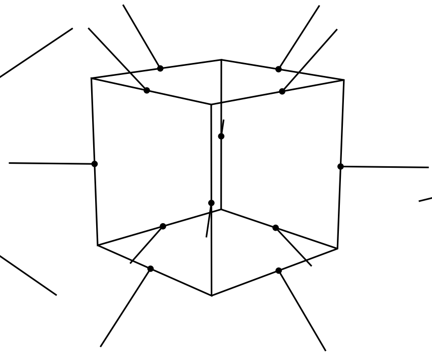
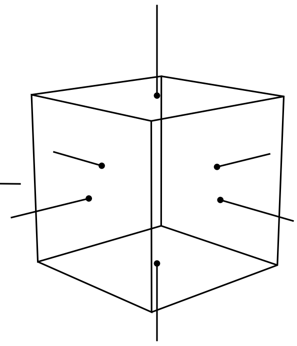



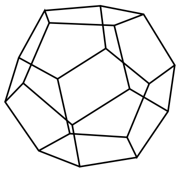
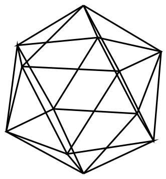
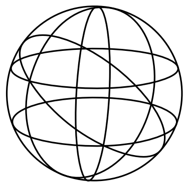
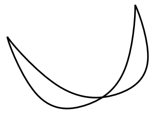




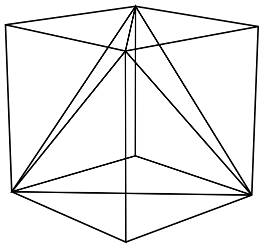
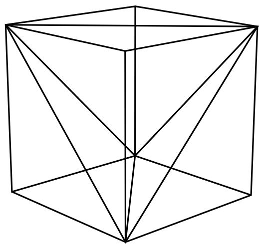
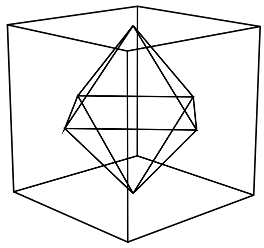
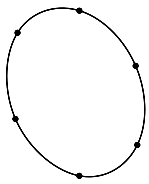





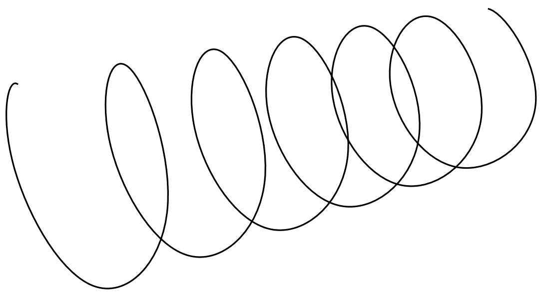
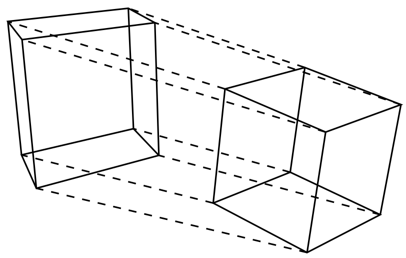





Setup. To obtain coordinates of the vertices of the regular polyhedra, consider a trapezoid inscribed in a unit circle, say with vertices of coordinates $(\pm r, l)$ and $(\pm r', l')$. If c is the common length of the two non-horizontal sides then

$$r^2 + l^2 = 1, \quad r'^2 + l'^2 = 1, \quad (r' - r)^2 + (l' - l)^2 = c^2.$$

Assume that $l' > l$ and $l' > 0$. Then, in terms of the two ratios $\mu = r/c$ and $\mu' = r'/c$, the coordinates are given by the formulas (where $L := 1 - (\mu - \mu')^2$ and $N := 1 + 4\mu\mu'$):

(1') $$c^2 = \frac{4L}{N}, \quad r^2 = \frac{4\mu^2 L}{N}, \quad r'^2 = \frac{4\mu'^2 L}{N},$$

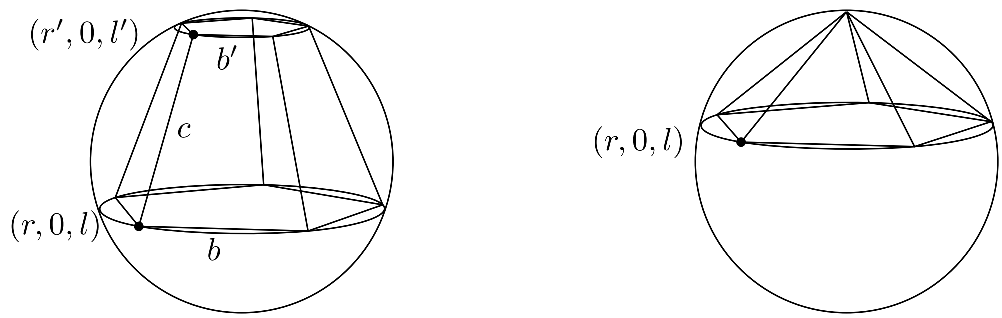
(1") $$l' - l = \frac{2L}{\sqrt{N}}, \quad l = \frac{-1 + 2\mu(\mu - \mu')}{\sqrt{N}}, \quad l' = \frac{1 + 2\mu'(\mu - \mu')}{\sqrt{N}}.$$

Note the degenerate case of a triangle: $r' := 0$, $l' = 1$. Then, $\mu' = 0$, and

(2) $$c^2 = 4(1 - \mu^2), \quad r^2 = 4\mu^2(1 - \mu^2), \quad l = -1 + 2\mu^2.$$

If r and r' are the radii of the circles circumscribing the base and the top of a regular truncated pyramid, based on a regular n-gon, and inscribed in a unit sphere, then the side lengths are $b = 2r \sin \pi/n$ and $b' = 2r' \sin \pi/n$. Therefore,

(3) $$\mu = \frac{b/c}{2 \sin \pi/n}, \quad \mu' = \frac{b'/c}{2 \sin \pi/n}.$$



The side lengths of the regular polyhedra. For example, by using (2) it is easy to determine the side length c of the regular polyhedra inscribed in a unit sphere, with a vertex at the North Pole:

For the tetrahedron, $n = 3$ and $b = c$. Hence $\mu^2 = 1/(2 \sin \pi/3) = 1/3$ and $c^2 = 8/3$.

For the hexahedron, $n = 3$ and $b = \sqrt{2}c$. Hence $\mu^2 = 2/(2 \sin \pi/3)^2 = 2/3$, and $c^2 = 4/3$.

For the octahedron, $n = 4$ and $b = c$. Hence $\mu^2 = 1/(2 \sin \pi/4)^2 = 1/2$, and $c^2 = 2$.

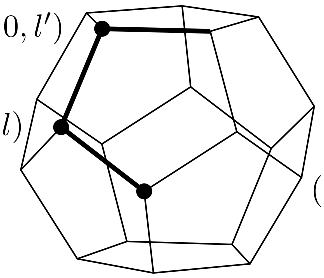
For the icosahedron, $n = 5$ and $b = c$. Hence $\mu^2 = 1/(2 \sin \pi/5)^2$. As $2 \sin^2 \pi/5 = 1 - \cos 2\pi/5 = (5 - \sqrt{5})/4$, it follows that $\mu^2 = 2/(5 - \sqrt{5}) = (5 + \sqrt{5})/10$, and $c^2 = 2(5 - \sqrt{5})/5$.

For the dodecahedron, $n = 3$, and b/c is the ratio between the diagonal and the side of regular pentagon, and hence equal to $(2 \sin 2\pi/5)/(2 \sin \pi/5) = 2 \cos \pi/5$.

So $\mu = \cos(\pi/5)/\sin(\pi/3)$ and $c^2 = 4(3-4\cos^2(\pi/5))/3 = 4(3-2(\cos(2\pi/5))/3 = (4\cos 2\pi/5)^2/3$.

Coordinates. For the icosahedron (one vertex on the North Pole), use (2) to determine the coordinates of the vertices: One vertex is $(r, 0, l)$, where r, l are given by (2): $r = 2/\sqrt{5}$, $l = 1/\sqrt{5}$.

For the dodecahedron (top face horizontal), use (1): $b/c = \sin(2\pi/5)/\sin(\pi/5) = 2\cos\pi/5$ and $b'/c = 1$. Hence $\mu = \cot\pi/5$ and $\mu' = 1/(2\sin\pi/5)$. It is fastest to use c determined above: $r = \mu c = (4/\sqrt{3})(\cos 2\pi/5)(\cot\pi/5)$, and $r' = \mu' c = (2/\sqrt{3})(\cos 2\pi/5)/(\sin\pi/5)$.




```
r1:=1/sqrt(3)*(sqrt(5)-1)/(2*sin(2*pi/10))
l1:=sqrt(1-r1*r1)
r:=1/sqrt(3)/sin(2*pi/10)
l:=sqrt(1-r*r)
```

The path marked above on the dodecahedron begins at the point x2 and ends at the point obtained by rotating x1 one fifth around the z-axis. So the path may be drawn by a PSline, produced by the following block of code:

```
beginps('PSline');
  cproj(x2);
  cproj(x);
  cproj(x1);
  rotateonefifth(x1);cproj(x1);
endps;
rotateonefifth(x2);rotateonfifth(x);
```

After the first 'rotateonefifth', the point x1 becomes the end point of the path. The next two 'rotateonefifth's do not contribute to the PSline, but when the code has been completed, all three points x2, x, x1 have been rotated. So, the top part of the dodecahedron may be drawn by repeating the code 5 times. The bottom part may be drawn by reflecting the top part in the origin.