

# Slicing Solids

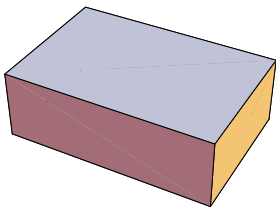
The package presented here creates animations that show the result of slicing a given graphics object by a sequence of parallel planes. In each frame, the two portions of the object on either side of the cross-section are shown side by side, as if on facing pages of a book. Viewing the animation is like flipping through the pages of the book, as the plane of the slices moves through the object.

Ross Moore

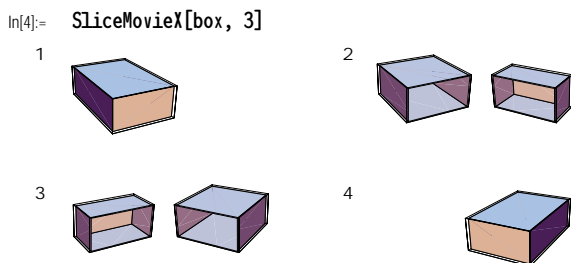
The package `SLiceMovie` is designed to look inside a `Graphics3D` object to see structure that might otherwise be hidden. It does this by “slicing through” the object and showing the two resulting parts side by side. Slicing can be done by a sequence of parallel planes perpendicular to the  $x$ ,  $y$ , or  $z$  axis, using the functions `SLiceMovieX`, `SLiceMovieY`, or `SLiceMovieZ`. For each plane, the functions create an animation frame showing the sliced-off portion on the right side and the remainder of the object on the left side.

The parallel cross-sections are like the pages of a book. Viewing the animation is similar to leafing through the book. As the animation progresses, the plane of the slice moves from front to back through the object, and the portion of the object shown on the right grows while the portion on the left shrinks.

```
In[1]:= Needs["SLiceMovie`"]
In[2]:= Needs["Graphics`Polyhedra`"]
In[3]:= box = Show[Polyhedron[Cube, {0,0,0}, {3,2,1}],
    Boxed -> False]
```

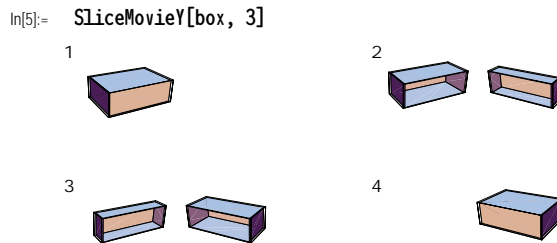


The first argument of the `SLiceMovie` functions is a `Graphics3D` object; the second argument gives the number of slices. This input creates four animation frames:



Ross Moore is a lecturer at Macquarie University, Australia. He obtained a D.Phil. from Oxford, England in 1981 studying mathematical physics and algebraic geometry. His research interests are in the use of mathematical software, especially *Mathematica*, *TeX*, and *PostScript*.

Here is the box sliced by planes perpendicular to the  $y$  axis.



The object can be sliced starting from the opposite end by setting the option `reverseSlice` to `True`. The option `frameRange` is used to generate a subsequence of the animation frames. The input `SLiceMovieX[graphic, n, frameRange -> {j, k}]` produces frames  $j$  through  $k$ , where  $j \leq k \leq n$ .

The `SLiceMovie` functions take the usual `Graphics3D` options, such as `Axes`, `AxesLabel`, `Boxed`, `LightSources`, and `ViewPoint`. If no `PlotRange` is specified, the `PlotRange` of the original graphics object is used. All other options of the graphics object are ignored.

Changes to the `ViewCenter` and `ViewVertical` options are possible, but they may upset the balanced perspective given by the default values; use `Options[SLiceMovieX]` to examine the settings of these options. For `SLiceMovieX` and `SLiceMovieY`, the default value for `ViewVertical` is the  $z$  axis, while for `SLiceMovieZ` it is the  $x$  axis, with coordinates decreasing vertically. The defaults for `ViewCenter` specify a corner of the bounding box at the face where slicing occurs. This setting is necessary (but not sufficient) for the slicing plane to appear fixed for all frames. Changing the option is recommended only with `Boxed` set to `False`, or else the object will appear to rotate or expand slightly as it is being sliced. Similarly, care should be taken with changes to `ViewPoint`. If axes are required, the `AxesEdge` setting may need to be adjusted to suit the `ViewPoint`.

## Maeder's Shell

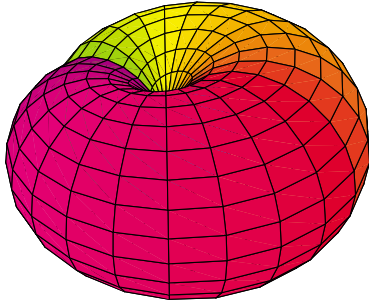
The following example uses the graphic from the cover of the first edition of Roman Maeder's excellent book *Programming in Mathematica*. (The code used here is slightly different from Maeder's; see [Maeder 1990, p. 222] or [Maeder 1991, p. 263].)

```
In[6]:= Needs["Graphics`ParametricPlot3D`"]
```

```

In[7]:= MaederShell = SphericalPlot3D[
  {Sin[theta] (2 + Cos[phi/2]),
   FaceForm[Hue[phi/(4 Pi)], Hue[phi/(4 Pi), 0.7, 1]]},
  {theta, 0.1, Pi - 0.1, Pi/24}, {phi, 0, 4Pi, Pi/12},
  PlotRange -> All, BoxRatios-> Automatic,
  Boxed -> False, Lighting -> False, Axes -> False ]

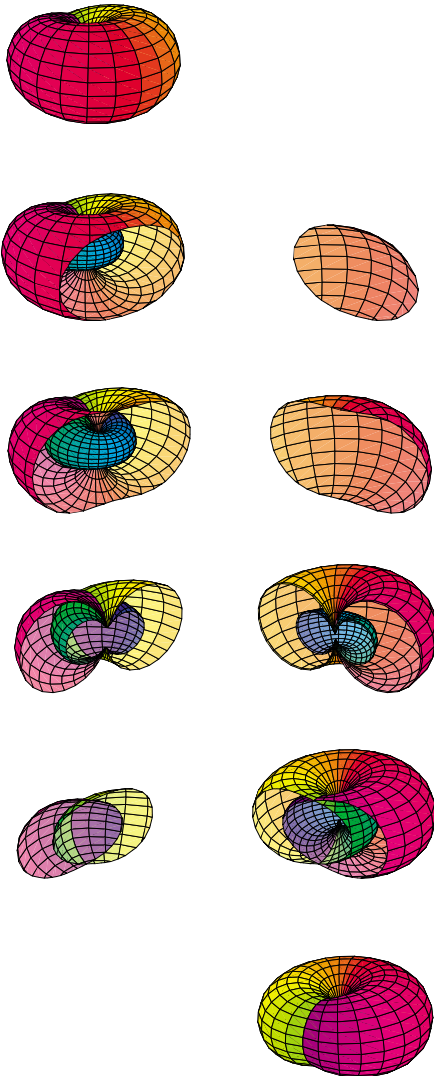
```



```

In[8]:= SliceMovieX[MaederShell, 5, Boxed -> False]

```



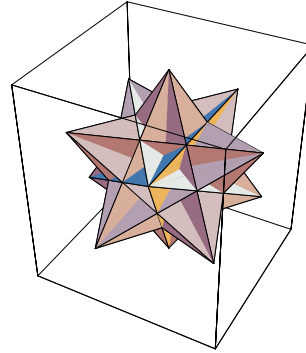
## The Great Icosahedron

Here is a slicing of the great icosahedron:

```

In[9]:= icoso = Show[Polyhedron[GreatIcosahedron]]

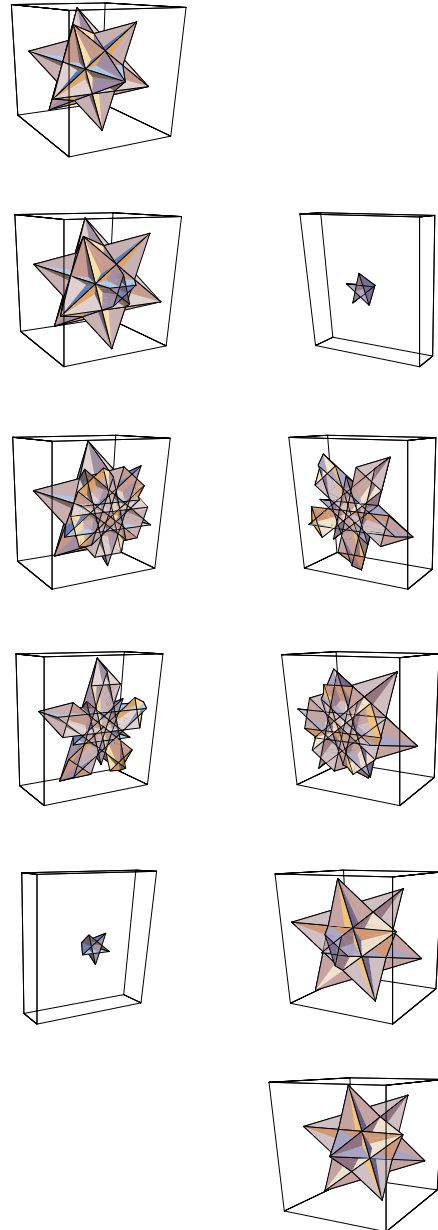
```



```

In[10]:= SliceMovieZ[icoso, 5]

```



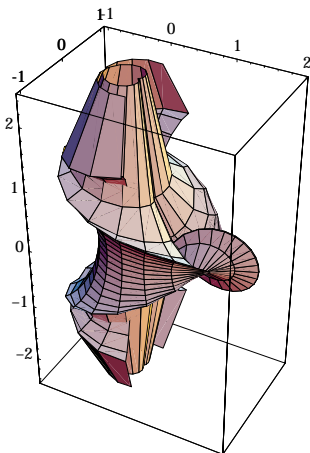
## Kuen's Surface

Kuen's surface has constant negative curvature. It is quite complicated, having a considerable amount of self-intersection. This is the standard parametrization of Kuen's surface [Gray 1993, p. 384]:

```
In[11]:= kuen[u_, v_] :=
  { 2(Cos[u] + u Sin[u])Sin[v]/(1 + u^2 Sin[v]^2),
    2(Sin[u] - u Cos[u])Sin[v]/(1 + u^2 Sin[v]^2),
    Log[Tan[v/2]] + 2 Cos[v]/(1 + u^2 Sin[v]^2) }
```

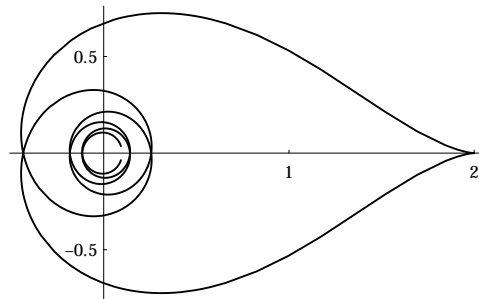
The parameter ranges are  $0 < v < \pi$  and  $-\infty < u < \infty$ . Normally the range for  $u$  is kept quite small (such as  $-4 < u < 4$ ) to avoid most of the self-intersection. We use a larger  $u$ -range here, specifically to show some of it.

```
In[12]:= kuensurface = ParametricPlot3D[
  kuen[u, v],
  {u, -6.5, 6.5}, {v, 0.01, Pi - 0.01},
  PlotPoints -> {30, 25},
  PlotRange -> {All, All, {-2.5, 2.5}},
  Ticks -> {Automatic, Range[-1, 1], Automatic} ]
```



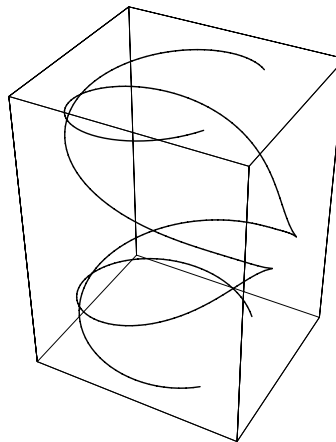
The surface crosses itself near  $u \sim \pm 4.5$ ; for larger  $|u|$  the surface remains within the shell established by lesser  $|u|$ . A similar crossing occurs for  $|u| \sim 7.7$  and near all solutions of  $u = \tan(u)$ . This behavior is most easily seen by plotting where  $v = \pi/2$  for a large range in  $u$ . This curve lies entirely within the plane  $z = 0$ .

```
In[13]:= ParametricPlot[
  Evaluate[Drop[kuen[u, Pi/2], -1]],
  {u, -20, 20},
  PlotRange -> All,
  Ticks -> {Range[2], Range[-0.5, 0.5]} ]
```



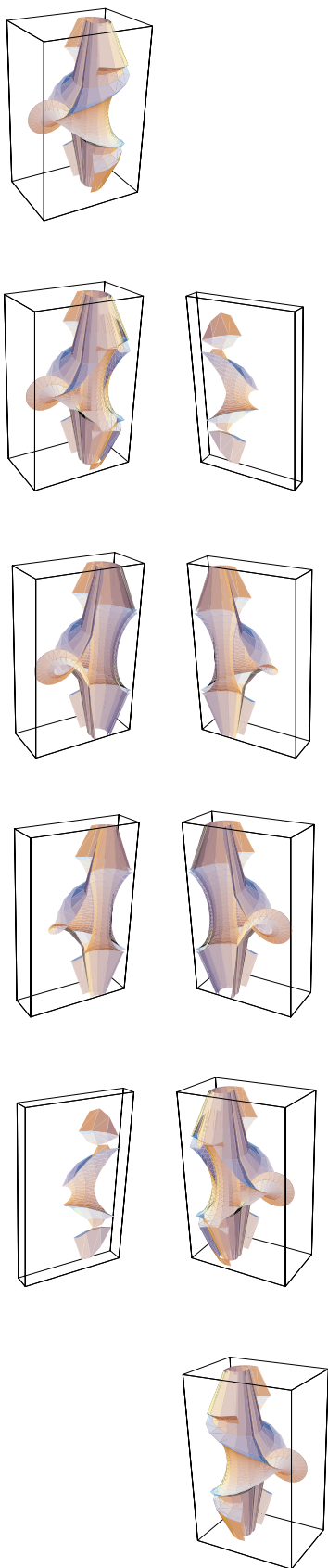
A sharp kink, or “cuspidal edge,” occurs in Kuen's surface along the locus where  $u \sin(v) = \pm 1$ .

```
In[14]:= ParametricPlot3D[ Evaluate[
  { kuen[u, ArcSin[1/u]], kuen[u, Pi - ArcSin[1/u]],
    kuen[-u, ArcSin[1/u]], kuen[-u, Pi - ArcSin[1/u]]}],
  {u, 1, 7},
  PlotRange -> All, Ticks -> None]
```



Here is a movie of parallel slices. If kuensurface is generated with large values in PlotPoints, then it is advisable to use the frameFile option to save the frames in separate files, for later rendering.

```
In[15]:= SliceMovieY[ Graphics3D[
  {EdgeForm[], kuensurface // First}],
  5,
  PlotRange -> PlotRange[kuensurface],
  ViewCenter -> {1, 1, 2/3}]
```



## Conserving Memory

Storing the graphics can take up lots of disk space, and viewing the movie can require a lot of memory for the *Mathematica* front end. To save memory, convert the images from PostScript to a bitmap format (PICT for Macintosh, XImage for the X front end) by choosing Convert from the Graph menu. The images should be scaled to the right size before they are converted, since bitmaps cannot be resized without affecting their quality.

If the graphics object is large and a lot of slices are required for the movie, the animation may exhaust the memory available to the front end renderer. It may be best to generate portions of the movie, using the `frameRange` option. Convert a sequence of frames to a bitmap format, after any resizing and realignment, and then generate more frames.

An alternative method is to send the PostScript for each frame directly to a separate file, using the `frameFile` option. The resulting files can be read, and the graphics converted to bitmaps, within a different notebook. This is perhaps the most space-efficient way to store the movie.


## Acknowledgments

Thanks go to Cameron Smith and Nancy Blachman for writing *The Mathematica Graphics Guidebook*. Without this book I would probably have taken much longer to appreciate the significance of changing the options `ViewCenter` and `ViewVertical`. A reading of their description, on pages 192–199, of the perspective transformations and coordinate systems used for the `ViewPoint` and bounding box in a 3D graphic has led to greater understanding of this aspect of *Mathematica* graphics. Particularly important was their description of the role played by the `BoxRatios` option in establishing these coordinate systems.

## References

- Gray, Alfred. 1993. *Modern Differential Geometry of Curves and Surfaces*. CRC Press.
- Smith, Cameron, and Nancy Blachman. 1995. *The Mathematica Graphics Guidebook*. Addison-Wesley.
- Maeder, Roman E. 1990. *Programming in Mathematica*. 1st. ed. Addison-Wesley.
- Maeder, Roman E. 1991. *Programming in Mathematica*. 2nd. ed. Addison-Wesley.

Ross Moore  
 Mathematics Department  
 Macquarie University  
 North Ryde, NSW 2109, Australia

 The electronic supplement contains the package `SLiceMovie.m` and the notebook `SLice Movie`.