

V3D: Portable interactive 3D graphics for PDF and beyond

John C. Bowman

Abstract

The V3D file format provides a portable, compressed representation for interactive 3D vector graphics generated by the $\text{\TeX}$ -aware vector graphics language Asymptote. It replaces the legacy PRC format, which was limited to Adobe Acrobat and lacked support for transparency, Bézier patches with vertex-dependent colors, and Bézier triangle patches. V3D content can be embedded in PDF files, viewed with an Okular plugin on GNU/Linux, or displayed online through a JavaScript-based viewer. We describe the format's features, demonstrate its capabilities through examples, and discuss its role in the ongoing modernization of Asymptote's rendering pipeline.

1 Introduction

The descriptive vector graphics language Asymptote¹ has since 2008 supported interactive 3D graphics embedded within PDF files. Until now, this has been accomplished using Adobe's compressed PRC format [3, 2, 1], which can describe smooth surfaces and curves without polygonal tessellation.

However, the PRC format has significant limitations. Most significantly, *no adaptive renderers* are available for PRC content. Even though it was introduced over two decades ago, PRC is still only viewable with the fixed-mesh proprietary renderer built into Adobe Acrobat, which is no longer supported on many platforms. The PRC specification lacks support for 3D transparency, Bézier triangle patches, and vertex-dependent colors for Bézier patches.

These limitations motivated the development of a new Vector 3D (V3D) format, following the specification and reference implementation available at github.com/vectorgraphics/v3d. The V3D format retains the advantages of a portable compact vector representation, while adding modern features and supporting multiple independent adaptive viewers.

2 The V3D format

V3D supports the following geometric primitives:

- Bézier patches (tensor-product cubic surfaces);
- Bézier triangles (expressed in barycentric coordinates);
- Geometric primitives: sphere, hemisphere, disk, cylinder, and tube;

- Triangle groups (tessellated meshes);
- 3D Bézier curves;
- 3D pixels.

In addition to geometry, V3D supports rendering features that were missing from PRC:

- 3D transparency with pixel-perfect order-independent compositing;
- Vertex-dependent colors (for palette-based gradient shading);
- Physically based rendering with image-based lighting.

3 Generating V3D output

V3D files can be generated from Asymptote source code in several ways.

To produce a standalone V3D file, use the command line `asy -f v3d file.asy` or the setting `settings.outformat="v3d"`.

To embed V3D content directly within an PDF file, use: `asy -f pdf -v3d file.asy`.

Manual embedding of a V3D file within a $\text{\LaTeX}$ document can be achieved either with the `asymptote` package:

```
\begin{asy}[width=0.95\linewidth,inline]
import label3solid;
\end{asy}
```

or the `media9` package:

```
\includemedi[a][noplaybutton,
activate=pagevisible,
width=401px,height=164px]{file.v3d}%
```

4 Viewers

One of the primary motivations for developing V3D was to break free from the single-viewer dependency imposed by PRC. The format now supports multiple independent viewers:

- **Asymptote's native viewer:** A V3D file can be viewed standalone with `asy -V file.v3d`, using Asymptote's built-in renderer.

V3D files can also be imported back into Asymptote in interactive mode:

```
import v3d;
importv3d("file.v3d");
```

- **Okular plugin:** A V3D plugin for the Okular document viewer enables interactive 3D viewing directly within PDF documents. Available on GitHub at [vectorgraphics/Okular-v3d-Plugins](https://github.com/vectorgraphics/Okular-v3d-Plugins), the plugin supports rotation, zooming, shifting, and panning of V3D content embedded in PDF files. It also supports viewing raw V3D files.

¹ Andy Hammerlindl, John Bowman, and Tom Prince, available under the GNU Lesser General Public License from [asymptote.sourceforge.net](https://sourceforge.net).

- **Online JavaScript viewer:** An online JavaScript-based V3D-aware PDF viewer available on GitHub at [vectorgraphics/pdfv3dReader](https://github.com/vectorgraphics/pdfv3dReader) allows 3D content to be viewed in any modern web browser without installing additional software.
- **Python library:** The `pyv3d` Python library provides programmatic access to V3D files, enabling conversion to alternate 3D formats and batch processing.

5 Examples

The remainder of this article demonstrates the capabilities of the V3D format through a series of examples. All figures are generated using Asymptote and embedded as V3D content within this PDF.

5.1 Lifting T_EX to 3D

The ability to typeset labels and equations with T_EX for overall document consistency is one of the distinguishing features of Asymptote. In three dimensions, glyph outlines are first partitioned into simply connected regions and then decomposed into planar Bézier surface patches using the `bezulate` algorithm [4]. These patches can then be extruded and arbitrarily transformed in three dimensions, as shown in Fig. 1.

$$\int_{-\infty}^{\infty} e^{-ax^2} dx = \sqrt{\pi/a}$$

Figure 1: A T_EX equation lifted to three dimensions using the `bezulate` algorithm and stored in V3D format.

Curved 3D labels are also supported, with text following the geometry of underlying surfaces (Fig. 2).

5.2 Vertex-dependent colors

V3D supports vertex-dependent colors, allowing surfaces to be shaded with color gradients that vary smoothly across the geometry. Figure 3 shows a surface colored by a palette representing the phase of the complex gamma function (red=0°, green=120°, blue=240°).

5.3 V3D transparency

Perhaps the most visually striking feature of V3D is its support for 3D transparency with order-independent compositing, as illustrated in Figs. 4–7.

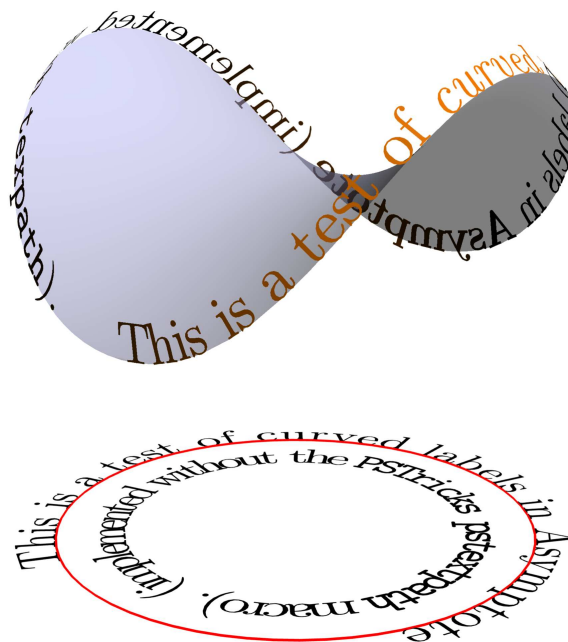


Figure 2: Curved 3D labels.

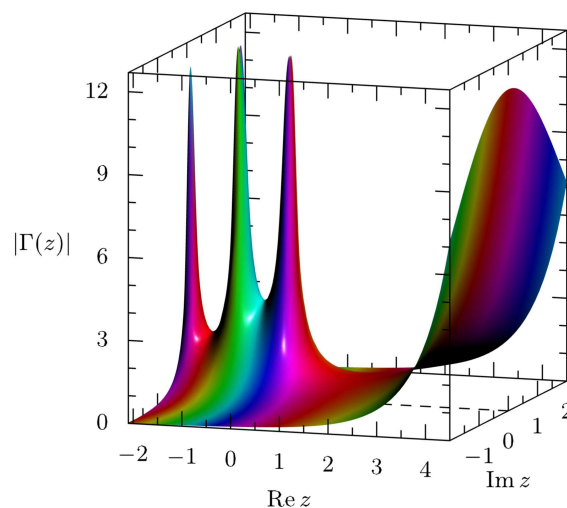


Figure 3: Vertex-dependent colors applied to a 3D surface.

6 The rendering pipeline

V3D is part of a broad modernization of Asymptote's rendering infrastructure. The legacy OpenGL renderer has been superseded by a Vulkan port, bringing order-independent transparency to all major platforms: macOS, Linux, and Microsoft Windows. Vulkan also provides better GPU utilization and more consistent rendering across operating systems.

For web deployment, the AsyGL library enables interactive 3D figures to be embedded directly in HTML files using WebGL rendering. This works on

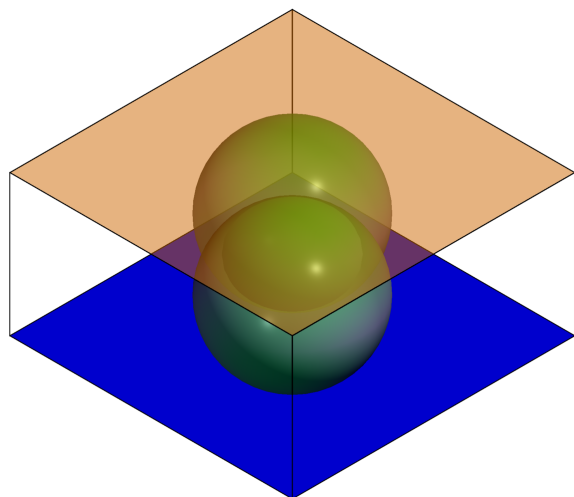


Figure 4: Two transparent spheres demonstrating order-independent transparency in V3D.

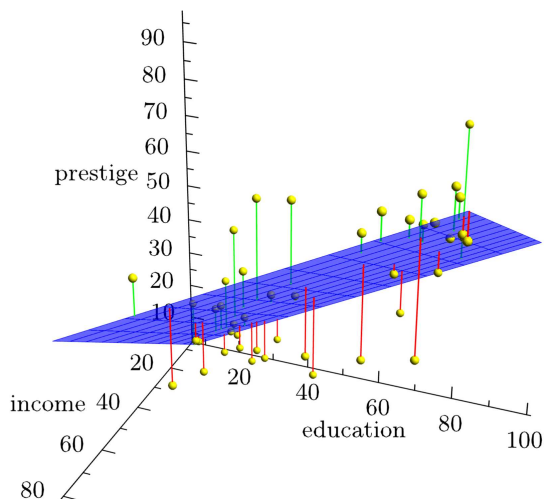


Figure 5: Linear regression visualization using a transparent plane.

tablets and phones and supports animations with user-controlled parameters via JavaScript. However, pixel-perfect order-independent transparency cannot be implemented, even in WebGL2, due to its lack of support for storage buffer objects (SSBOs).

This gap will be addressed by the upcoming migration from WebGL to WebGPU, bringing order-independent transparency to web browsers along with compute shader support and improved performance on modern GPUs.

7 Image-based lighting

Asymptote supports image-based lighting with `settings.ibl=true`, which uses pre-rendered EXR environment maps to produce physically based rendering with accurate reflections. For WebGL ren-

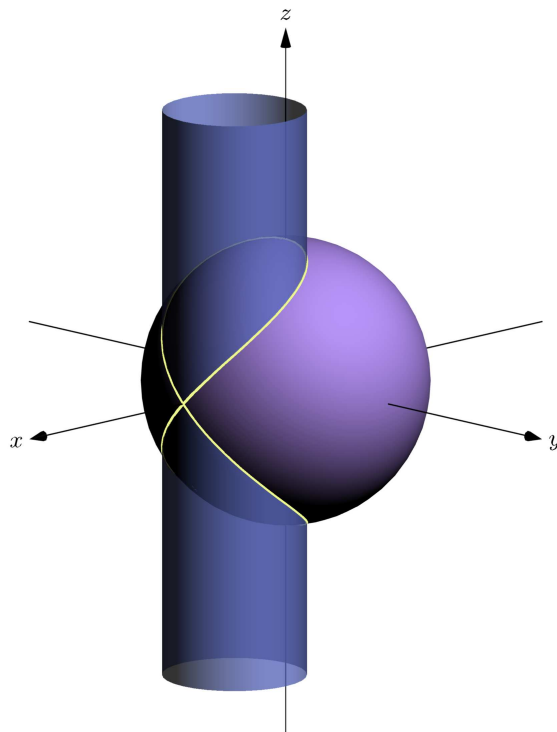


Figure 6: Viviani's curve obtained by intersecting a transparent cylinder with a sphere.

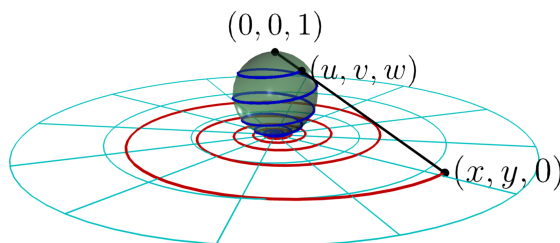


Figure 7: The projection of the extended complex plane onto the Riemann sphere.

dering, these maps are loaded from the network; for native rendering, they are read from a local directory. Additional environment maps can be generated on an NVIDIA GPU using the `reflect` program in the `cudareflect` subdirectory of the Asymptote source distribution. An example is shown in Fig. 8.

8 Bézier surfaces

For rendering surfaces, V3D supports both cubic Bézier triangles and Bézier patches.

Bézier triangles are the natural generalization of Bézier curves to surfaces, just as triangles are the natural generalization of straight lines to planar surfaces. They can be conveniently expressed in barycentric coordinates. Asymptote renders them



Figure 8: A Bézier triangle approximation to the unit sphere rendered with image-based lighting.

with an efficient in-house adaptive algorithm based on equilateral subdivision.

A useful application of Bézier triangles is for rendering a high-performance approximation to the unit sphere with a uniform mesh, as shown in Fig. 8 (the Bézier triangles and rendered wireframe mesh can be viewed with the Okular plugin by pressing the m key).

Bézier triangles are also used by Charles Staats' `smoothcontour3` module for drawing smooth implicit surfaces like the one shown in Fig. 9.

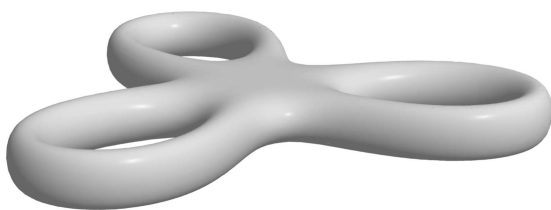


Figure 9: A genus-three surface.

Bézier patches (tensor-product cubic surfaces) are the natural generalization of quadrilaterals to nonplanar surfaces. They are well suited to general surface rendering, being the direct product of two Bézier curves.

9 Conclusion

The V3D format represents a significant step forward in the portability and capabilities of interactive 3D graphics generated by Asymptote. By replacing the PRC format with an open, multi-viewer alternative,

it ensures that Asymptote's 3D output remains accessible across platforms and software ecosystems.

The addition of transparency, vertex-dependent colors, Bézier triangle support, and physically based rendering brings V3D in line with modern graphics standards. Together with the Vulkan renderer and the AsyGL library for web deployment, these advances position Asymptote to continue its evolution as a vector graphics language designed to complement $\text{\TeX}$ for technical illustration.

Future work includes the migration from WebGL to WebGPU (to bring order-independent transparency to browsers), V3D animations, 3D clipping, and improved accessibility support.

Acknowledgments

I would like to thank Andy Hammerlindl for designing and implementing much of the underlying Asymptote language, Orest Shardt and Jesse Frohlich for their exceptional work on Bézier surface algorithms, Michail Vidiassov for his PRC driver overhaul, Olivier Guibé for splined parametric surfaces, Jamie Rasmameemasuang for their assistance in implementing V3D, physically based rendering, image-based lighting, and the AsyGL library, Charles Staats for templated imports and the `smoothcontour3` module, and Ben Bingham for the Okular plugin. Financial support for this work was provided by the Natural Sciences and Engineering Research Council of Canada.

References

- [1] J.C. Bowman. Asymptote: Interactive $\text{\TeX}$ -aware 3D vector graphics. *TUGboat* 31(2):203–205, 2010. tug.org/TUGboat/tb31-2/tb98bowman.pdf
- [2] J.C. Bowman, O. Shardt. Asymptote: Lifting $\text{\TeX}$ to three dimensions. *TUGboat* 30(1):58–63, 2009. tug.org/TUGboat/tb30-1/tb94bowman.pdf
- [3] ISO/TC171/SC2. 3D use of product representation compact (PRC) format. [en.wikipedia.org/wiki/PRC_\(file_format\)](http://en.wikipedia.org/wiki/PRC_(file_format))
- [4] O. Shardt, J.C. Bowman. Surface parametrization of nonsimply connected planar Bézier regions. *Computer-Aided Design*, 44(5):484.e1–10, May 2012. www.math.ualberta.ca/~bowman/publications/cad10.pdf

◇ John C. Bowman
Dept. of Mathematical and Statistical
Sciences
University of Alberta
Edmonton, Alberta T6G 2G1
Canada
[bowman \(at\) math dot ualberta dot ca](mailto:bowman@math.ualberta.ca)
www.math.ualberta.ca/~bowman/