BUILDING MATHEMATICAL STRUCTURES IN MINETEST, AN OPEN-SOURCE VOXEL GAME

Kyle M. Claassen
Department of Mathematics
Rose-Hulman Institute of Technology
Terre Haute, IN 47803, USA
claassen@rose-hulman.edu

BUILDING MATHEMATICAL STRUCTURES IN MINETEST, AN OPEN-SOURCE VOXEL GAME

Abstract: Minetest is a free, open-source voxel game similar to Minecraft in which players build arbitrary structures by placing individual cubical blocks. Minetest has superior potential as an educational tool in the classroom, and due to its easily accessible modding API, the game's three-dimensional world can be used a canvas for plotting mathematical curves, surfaces, and solids. This is a fun and relatable way to empower students at various levels of maturity to (literally!) shape their world with mathematics.

Keywords: visualization, educational technology, games, active learning, interactive plotting, curves and surfaces, multivariable calculus, triple integrals, ODE systems, open source, Minetest, Minecraft

Years ago when I was still in graduate school, my brother introduced me to *Minecraft*, a popular commercial computer game in which players build arbitrary structures by placing individual cubical blocks in a virtual three-dimensional world. During my first serious encounter with this game, a few thoughts occurred to me:

1. This is taking too long.

Due to the low resolution of the blocky world, intricate structures must be rather large in size, much like how a digital photograph looks "pixelated" unless the dimensions of the image are large. It is very tedious and time-consuming to build gigantic structures by placing one block at a time, and I am impatient.

2. It would be convenient to generate structures based on mathematical equations.

For example, a sphere is defined by a very simple equation, but it is rather difficult to manually build one in the game. A "sphere" would need be large in order to adequately approximate its curvature, and placing the blocks precisely would require very careful planning. It is easy to plot a sphere in common mathematics software, so why shouldn't it be similarly easy to plot a sphere (or any other mathematical curve or surface) in the game?

3. This would be fun to use in the classroom!

Rather than always producing plots with standard software (such as a computer algebra system), it would be novel to demonstrate curves and surfaces as impressive Minecraft structures. Moreover, many students would likely be familiar with the game already and would appreciate the difficulty of building such structures. Who knows, they might even embrace the relevant mathematics!

1 MINETEST: AN OPEN-SOURCE MINECRAFT ALTERNATIVE

Generating structures based on equations requires interacting with the world programmatically. Though it is technically possible to create Minecraft "mods" that extend and modify the gameplay, it is not especially inviting to do so; Minecraft modding is not officially supported, and I was intimidated by complexity of getting started. Before going down that rabbit hole, I discovered that Minecraft is a specific example of a *voxel game*. (A "voxel" is a higher-dimensional analog of a "pixel", i.e. the discrete unit of space in a two-dimensional image.) A survey of alternative voxel games led me to *Minetest* [1], a free/open-source voxel game engine that features gameplay similar to that of Minecraft. Though Minetest is not as well-known, it has greatly superior potential as an educational tool:

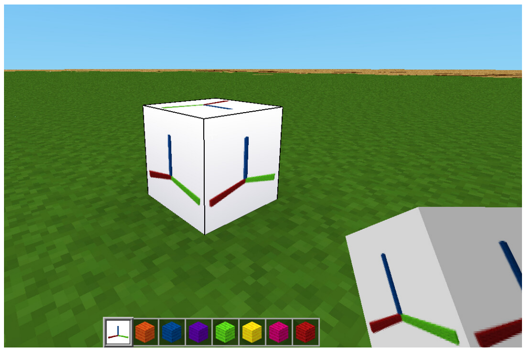
- Minetest is free to download and play.
- Minetest has modest hardware requirements and runs on many popular operating systems.
- Most importantly for me, **Minetest is intentionally designed to be extended by the open-source community.**

Minetest gameplay can be extended via mods written in the Lua scripting language [3], and developing Minetest mods is highly encouraged. Its API (Application Programming Interface) for performing game operations programmatically has excellent documentation for both beginners [8] and experts [2], and there is a central repository of open-source mods [7] that one can refer to for inspiration and instruction. Moreover, one only needs a text editor to write the code for a Minetest mod; there is no requirement to install special software or engage in sophisticated software development processes, so it's easy to get started even if you only have a modest computer programming background.

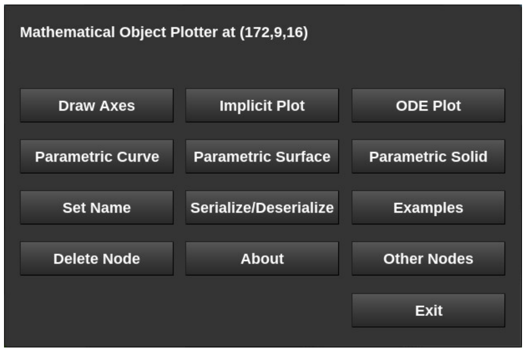
Ultimately, I have developed *MathPlot* [4], a Minetest mod for interactively plotting parametric and implicit curves/surfaces/solids, as well as solution trajectories for systems of ordinary differential equations. In what follows, I will provide a high-level[1] overview of

[1] A detailed discussion of how to use the various features of the MathPlot mod is beyond the scope of this work, but more information about how to install and use the mod can be found on the MathPlot project wiki [5].

Figure 1: MathPlot Origin Node



(a) An origin node that defines the location of $(x, y, z) = (0, 0, 0)$.



(b) A MathPlot origin node's main menu.

MathPlot's capabilities and how it can be used to empower students at various levels of maturity to shape their world with mathematics.

2 PLOTTING CURVES AND SURFACES

The Minetest voxel world is a novel setting to explore many ideas that arise in multivariable calculus, especially plotting curves and surfaces. In the voxel world, each block is a $1 \times 1 \times 1$ cube whose location is specified in rectangular coordinates[2]. The MathPlot mod provides functionality for plotting implicit surfaces defined by equations of the form $f(x, y, z) = 0$, as well as curves/surfaces defined by parametric equations. The expressions are specified in the syntax of the Lua programming language [3], which is very similar to standard line-based input for calculators.

The MathPlot mod provides a block called an "Origin Node", which defines the position in the Minetest world that corresponds to $(x, y, z) = (0, 0, 0)$ in the plots; see Figure 1a. Punching an origin node displays a menu of the various operations that can be performed (see Figure 1b). As an example, the ellipsoid defined by

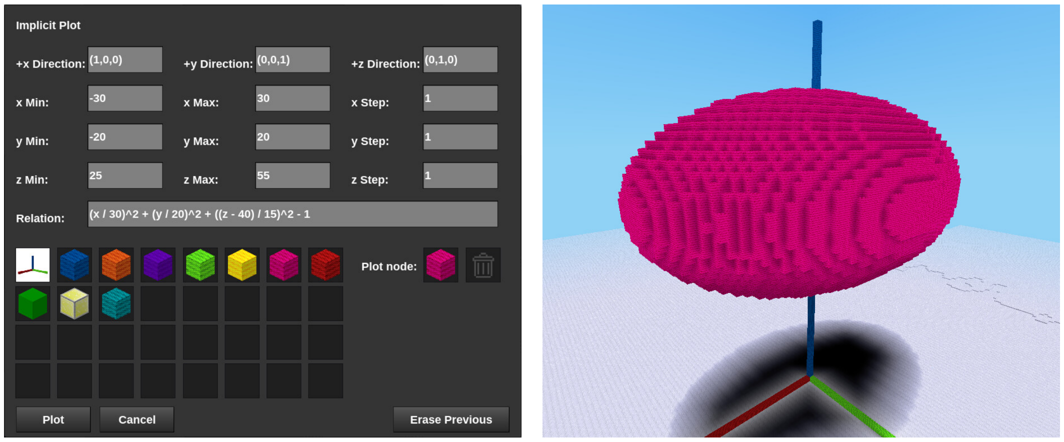
$$\left(\frac{x}{30}\right)^2 + \left(\frac{y}{20}\right)^2 + \left(\frac{z-40}{15}\right)^2 - 1 = 0$$
$$\text{on } -30 \le x \le 30,\ -20 \le y \le 20,\ 25 \le z \le 55$$

can be plotted using the `Implicit Plot` screen, which is used to plot points (x, y, z) that satisfy an equation of the form $f(x, y, z) = 0$. The left-hand side of the equation is input into the `Relation` field (the "$= 0$" is implied), the x-, y-, and z-ranges are specified, and the type of block which will be used to draw the surface (the "plot node") is chosen from the player's inventory; see Figure 2.

Parametric curves and surfaces can be plotted as well. For example, using the `Parametric`

[2]As is relatively common in computer graphics, the Minetest engine uses a left-handed coordinate system in which the positive y-axis is in the upward direction. However, the MathPlot mod compensates so that, by default, plots are performed in the usual right-handed coordinate system where the positive z-axis is in the upward direction.

Figure 2: An ellipsoid created using the `Implicit Plot` screen.



Implicit Plot

+x Direction: (1,0,0) +y Direction: (0,0,1) +z Direction: (0,1,0)

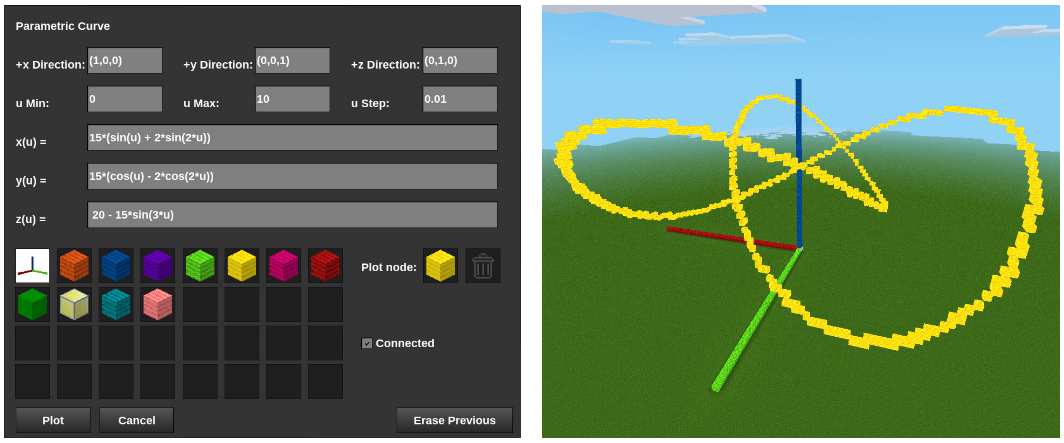
x Min: -30 x Max: 30 x Step: 1

y Min: -20 y Max: 20 y Step: 1

z Min: 25 z Max: 55 z Step: 1

Relation: (x / 30)^2 + (y / 20)^2 + ((z - 40) / 15)^2 - 1

Figure 3: A Trefoil Knot created using the `Parametric Curve` screen.



Parametric Curve

+x Direction: (1,0,0) +y Direction: (0,0,1) +z Direction: (0,1,0)

u Min: 0 u Max: 10 u Step: 0.01

x(u) = 15*(sin(u) + 2*sin(2*u))

y(u) = 15*(cos(u) - 2*cos(2*u))

z(u) = 20 - 15*sin(3*u)

[✓] Connected

`Curve` screen, one can plot a Trefoil Knot [12] parametrized by

$$\begin{cases} x(u) = 15\,(\sin(u) + 2\sin(2u)) \\ y(u) = 15\,(\cos(u) - 2\cos(2u)) \\ z(u) = 20 - 15\sin(3u) \quad \text{on} \quad 0 \le u \le 2\pi, \end{cases}$$

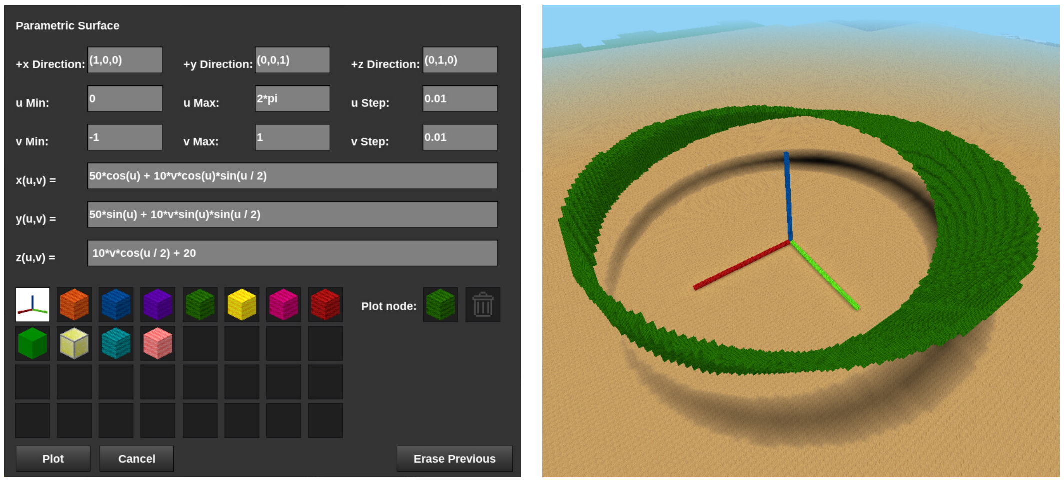
which is shown in Figure 3. Similarly, using the `Parametric Surface` screen, one can plot a large Möbius Strip [11] parametrized by

$$\begin{cases} x(u,v) &= 50\cos(u) + 10v\cos(u)\sin\left(\frac{u}{2}\right) \\ y(u,v) &= 50\sin(u) + 10v\sin(u)\sin\left(\frac{u}{2}\right) \\ z(u,v) &= 10v\cos\left(\frac{u}{2}\right) + 20 \quad \text{on } 0 \le u \le 2\pi, \quad -1 \le v \le 1, \end{cases}$$

which is shown in Figure 4.

It is also possible to plot solution trajectories for systems of ordinary differential equa-

Figure 4: A Möbius Strip created using the `Parametric Surface` screen.



Parametric Surface

+x Direction: (1,0,0) +y Direction: (0,0,1) +z Direction: (0,1,0)

u Min: 0 u Max: 2*pi u Step: 0.01

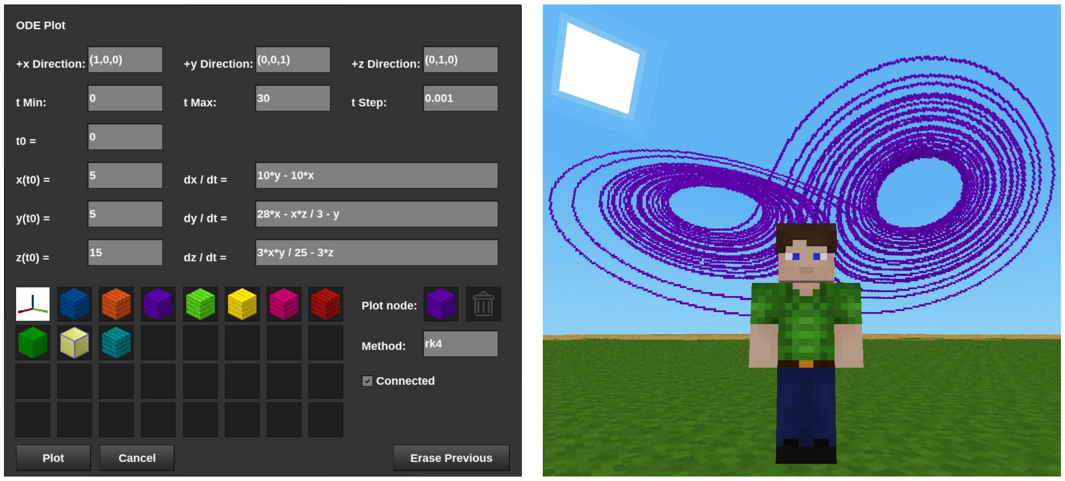
v Min: -1 v Max: 1 v Step: 0.01

x(u,v) = 50*cos(u) + 10*v*cos(u)*sin(u / 2)

y(u,v) = 50*sin(u) + 10*v*sin(u)*sin(u / 2)

z(u,v) = 10*v*cos(u / 2) + 20

Figure 5: A Lorenz Attractor created using the `ODE Plot` screen.



ODE Plot

+x Direction: (1,0,0) +y Direction: (0,0,1) +z Direction: (0,1,0)

t Min: 0 t Max: 30 t Step: 0.001

t0 = 0

x(t0) = 5 dx / dt = 10*y - 10*x

y(t0) = 5 dy / dt = 28*x - x*z / 3 - y

z(t0) = 15 dz / dt = 3*x*y / 25 - 3*z

Method: rk4

☑ Connected

tions using the `ODE Plot` screen (see Figure 5), such as a Lorenz Attractor [10]:

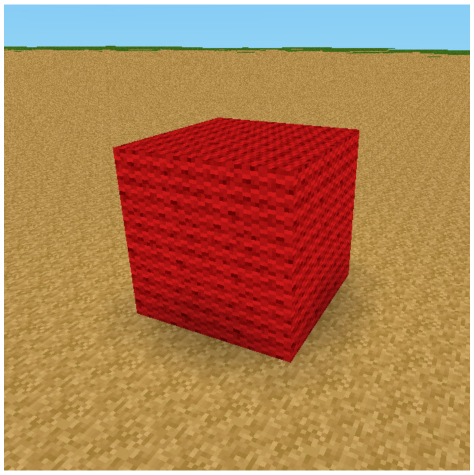
$$\begin{cases} \dfrac{dx}{dt} = 10y - 10x \\ \dfrac{dy}{dt} = 28x - \dfrac{1}{3}xz - y \\ \dfrac{dz}{dt} = \dfrac{3}{25}xy - 3z \quad \text{on } 0 \le t \le 30 \text{ with } x(0) = y(0) = 5,\ z(0) = 15. \end{cases}$$

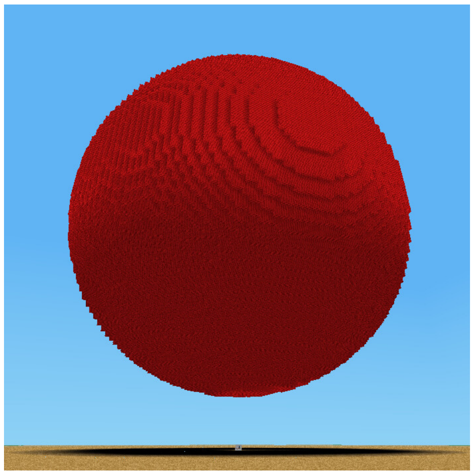
Two-dimensional ODE systems in the xy-plane can be plotted by setting $dz/dt = 0$ and using $z(t_0) = 0$ in the initial data.

3 THE IMPORTANCE OF SCALING

All the examples that have been presented so far have one important thing in common: they are *big*. This is necessary in order to see structural details due to the low resolution of the voxel world, and equations that are more appropriate for plotting large objects can be obtained by transforming typical "small-scale" equations that are commonly encountered

Figure 6: The plot for a sphere of radius 1 vs. the plot for a sphere of radius 50.





in mathematics classes. In particular, affine transformations of the form

$$x \mapsto \frac{x - T_x}{S_x}, \quad y \mapsto \frac{y - T_y}{S_y}, \quad z \mapsto \frac{z - T_z}{S_z} \tag{1}$$

are especially useful, where a translation by (T_x, T_y, T_z) follows a scale change with factors S_x, S_y, and S_z. Such transformations are often taught at the precalculus level for adjusting the dimensions and positions of function graphs and common implicit surfaces such as spheres/ellipsoids, and Minetest provides an excellent setting (with a very practical motivation!) for using these transformations.

Extremely simple examples reveal the need for rescalings. For instance, the resolution of the world is insufficient to produce a nice plot of the unit sphere $x^2 + y^2 + z^2 - 1 = 0$. However, rescaling all axes by a factor of 40 and translating by 50 units in the positive z-direction yields the equation

$$\left(\frac{x}{40}\right)^2 + \left(\frac{y}{40}\right)^2 + \left(\frac{z - 50}{40}\right)^2 - 1 = 0,$$

which produces a much more impressive plot (see Figure 6). Of course, one can apply (1) to parametric equations, which transforms a parametric curve $u \mapsto (x(u), y(u), z(u))$ into $u \mapsto (S_x \cdot x(u) + T_x,\ S_y \cdot y(u) + T_y,\ S_z \cdot z(u) + T_z)$. For example, in Figure 3, a uniform scaling $S_x = S_y = S_z = 15$ and a vertical translation with $T_z = 20$ was applied to the equations given in [12].

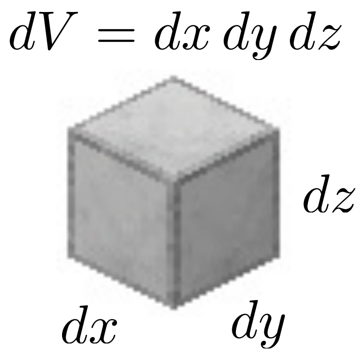
4 CLASSROOM USES

Hopefully you're convinced that it would be a lot of fun for students (and instructors!) to explore mathematical concepts using Minetest. For further inspiration, here are some concrete examples of how I have used Minetest with the MathPlot mod in educational settings.

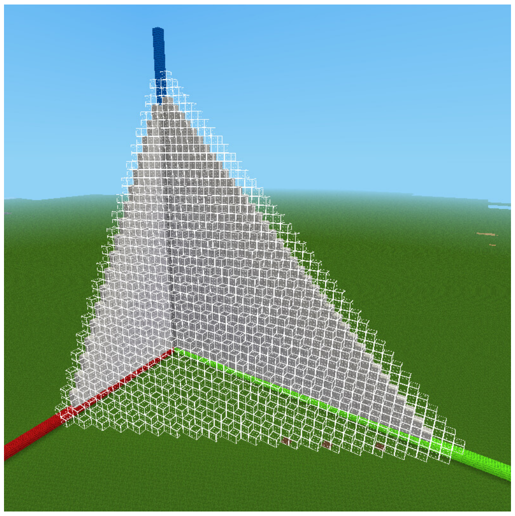
4.1 Explaining Triple Integrals

All structures in the game consist of cubical blocks, which fosters the fundamental notion that a solid region in $\mathbb{R}^3$ can be approximated by a union of small, discrete rectangular

Figure 7: Approximating the solid tetrahedron W using "small blocks"



(a) For visualization purposes, the volume of an individual block can be regarded as "dV".



(b) The approximate tetrahedron, W.

prisms. Of course, this is exactly how I want my students to visualize regions of integration, whose volume (or mass, etc.) can be computed as a summation of "small blocks", each having volume dV (see Figure 7a). That is,

$$\text{Volume}(W) = \iiint_W dV.$$

Computing such an integral explicitly usually requires an iterated integral, and Minetest is an excellent environment for demonstrating the thought process behind setting up various orders of integration!

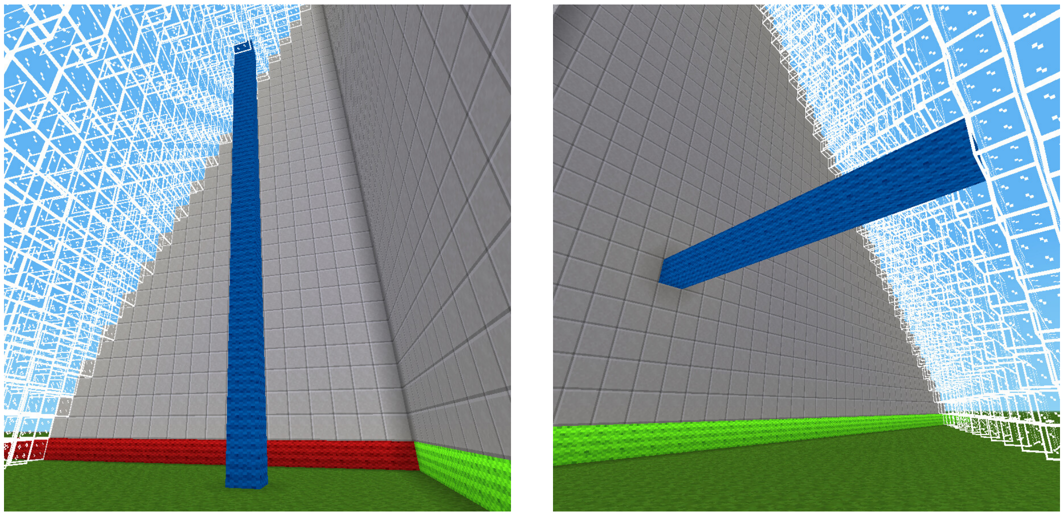
When teaching Calculus III, my first example of a triple integral is to find the volume of the solid tetrahedron W in the first octant bounded by the plane $2x + y + z = 40$ as shown in Figure 7b[3]. One can build W by first selecting a specific point (x, y) in the "floor" at $z = 0$, i.e. $(x, y) \in \{(x, y) : 0 \le x \le 20, 0 \le y \le 40 - 2x\}$, then "stacking blocks vertically" until reaching the "ceiling" at $z = 40 - 2x - y$. (See Figure 8a.) Keeping a running total of the blocks' volumes (i.e. the integral) yields the volume of W, and this process naturally describes the following iterated integral:

$$\text{Volume}(W) = \underbrace{\int_{x=0}^{20}\int_{y=0}^{40-2x}}_{\substack{\text{select a point } (x,y) \\ \text{on the floor...}}} \underbrace{\int_{z=0}^{40-2x-y}}_{\text{...then stack blocks vertically!}} dz\,dy\,dx = \frac{16000}{3} \text{ cubic units.}$$

Of course, one could compute the volume of W by a different iteration, e.g. building W by first selecting a block on the "back wall" in the yz-plane, i.e. $(y, z) \in \{(y, z) : 0 \le y \le 40, 0 \le z \le 40 - y\}$, then "stacking blocks forward" perpendicular to the yz-plane from $x = 0$ to $x = (40 - y - z)/2$. (See Figure 8b.) This process describes a different iterated

[3]The walls of the tetrahedron can be built quickly using the `Implicit Plot` screen.

Figure 8: Columns of blocks



(a) A vertical column of blocks perpendicular to the xy-plane, per the setup of an iterated integral in the order $dz\,dy\,dx$ or $dz\,dx\,dy$.

(b) A column of blocks perpendicular to the yz-plane, per the setup of an iterated integral in the order $dx\,dy\,dz$ or $dx\,dz\,dy$.

integral, which of course yields the same volume:

$$\text{Volume}(W) = \underbrace{\int_{y=0}^{40}\int_{z=0}^{40-y}}_{\substack{\text{select a point } (y,z)\\ \text{in the back wall...}}} \underbrace{\int_{x=0}^{(40-y-z)/2}}_{\text{...then stack blocks forward!}} dx\,dz\,dy = \frac{16000}{3} \text{ cubic units.}$$

Indeed, based on direct student feedback (they love seeing Minetest in class), I have found that showing the "physical" act of stacking blocks in Minetest helps make the process of setting up iterated triple integrals much more tangible.

4.2 Prototyping with a Computer Algebra System

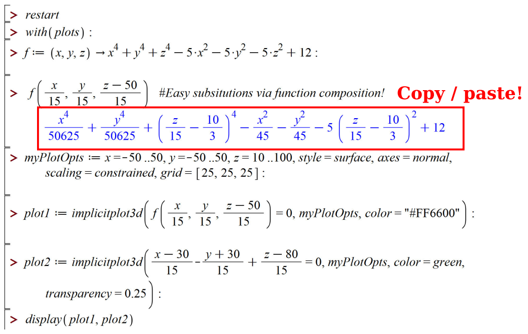
The equations for curves and surfaces can become complicated (and error-prone) very quickly, so it can be beneficial to prototype the structures in a computer algebra system before plotting them in Minetest. This is an excellent opportunity to practice using general-purpose mathematics software, as the code/worksheet provides a convenient record of the equations and parameter ranges used to generate the plots, which can then be transcribed into Minetest. Moreover, mistakes in equations can cause accidental damage to other structures in your Minetest world, so it is helpful to confirm that multiple curves/surfaces fit together as expected before plotting them in the game. Also, it may be possible to copy/paste complicated equations from the computer algebra system directly into Minetest with relatively few modifications. Indeed, copy/paste compatibility with the Maple computer algebra system is nearly perfect; one very rarely encounters errors when pasting Maple's symbolic output into the MathPlot equation fields.

To showcase the convenience of prototyping with a CAS in Calculus III, as an in-class example I demonstrate how to plot and transform a *Tanglecube* [9], an implicit surface defined by

$$f(x, y, z) := x^4 + y^4 + z^4 - 5x^2 - 5y^2 - 5z^2 + 12 = 0. \tag{2}$$

It is difficult to visualize the Tanglecube surface based on (2), so it is helpful to use a computer algebra system to plot it and experiment with various window ranges for x, y, and z. Moreover, one can easily verify that $(x, y, z) = (2, -2, 2)$ is on this surface and that $\vec{n} = \nabla f(2, -2, 2) = \langle 12, -12, 12\rangle$ is a suitable normal vector to the surface at that point, hence the tangent plane is given by $12(x-2) - 12(y+2) + 12(z-2) = 0$. Of course, our goal

Figure 9: Prototyping a surface and its tangent plane in Maple, then plotting in Minetest



> $restart$

> $with(plots):$

> $f := (x, y, z) \rightarrow x^4 + y^4 + z^4 - 5\cdot x^2 - 5\cdot y^2 - 5\cdot z^2 + 12 :$

> $f\left(\frac{x}{15}, \frac{y}{15}, \frac{z-50}{15}\right)$ *#Easy subsitutions via function composition!* **Copy / paste!**

$$\boxed{\frac{x^4}{50625} + \frac{y^4}{50625} + \left(\frac{z}{15} - \frac{10}{3}\right)^4 - \frac{x^2}{45} - \frac{y^2}{45} - 5\left(\frac{z}{15} - \frac{10}{3}\right)^2 + 12}$$

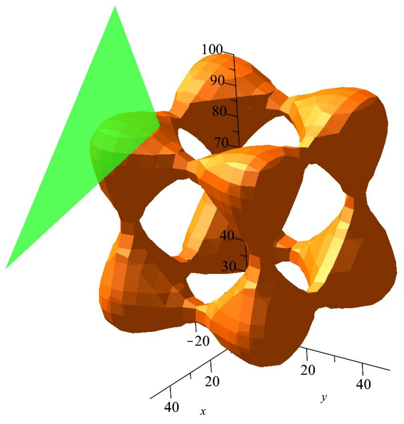
> $myPlotOpts := x = -50..50, y = -50..50, z = 10..100, style = surface, axes = normal,$
> $scaling = constrained, grid = [25, 25, 25] :$

> $plot1 := implicitplot3d\left(f\left(\frac{x}{15}, \frac{y}{15}, \frac{z-50}{15}\right) = 0, myPlotOpts, color = \text{"\#FF6600"}\right):$

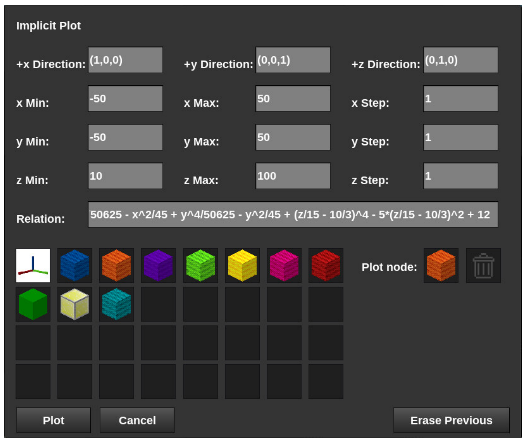
> $plot2 := implicitplot3d\left(\frac{x-30}{15} - \frac{y+30}{15} + \frac{z-80}{15} = 0, myPlotOpts, color = green,$
> $transparency = 0.25\right):$

> $display(plot1, plot2)$

(a) It is quick to experiment with transformed equations and plots in Maple through function composition or substitutions.



(b) The prototyped plot in Maple.



Implicit Plot

+x Direction: (1,0,0) +y Direction: (0,0,1) +z Direction: (0,1,0)

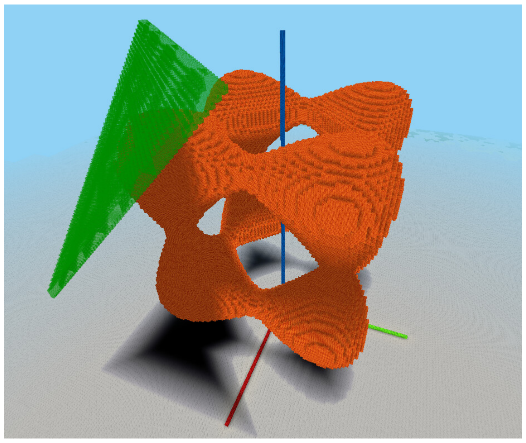
x Min: -50 x Max: 50 x Step: 1

y Min: -50 y Max: 50 y Step: 1

z Min: 10 z Max: 100 z Step: 1

Relation: 50625 - x^2/45 + y^4/50625 - y^2/45 + (z/15 - 10/3)^4 - 5*(z/15 - 10/3)^2 + 12

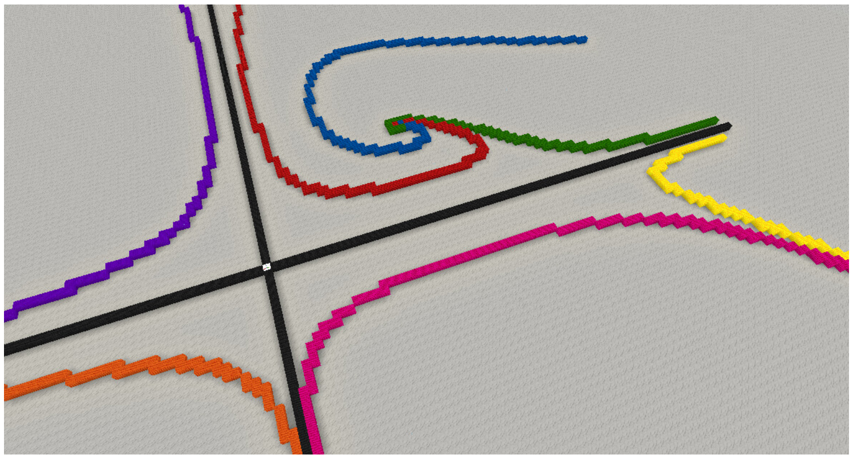
(c) The transformed equation in Maple can be directly copied and pasted into Minetest.



(d) The resulting plot in Minetest.

will be to plot the Tanglecube and its tangent plane in Minetest! As previously discussed, the equations must be modified to scale and translate the plots so that they look good in the game. The Tanglecube's equation (2) has many instances of x, y, and z, hence it can be tedious to manually implement and experiment with the transformations. However, this is easy to do in a CAS such as Maple, and the resulting expression (which is much more complicated) can be copied from Maple and pasted directly into the `Relation` field

Figure 10: A phase portrait created with the `ODE Plot` screen.



on the `Implicit Plot` screen. With some experimentation, one finds that stretching all axes by a factor of 15 and translating the Tanglecube vertically by 50 units yields a nice plot in Minetest on $-50 \leq x, y \leq 50$, $10 \leq z \leq 100$. See Figure 9 for a comparison of the prototyped plot in Maple and the corresponding plot in Minetest.

4.3 Drawing Phase Portraits of ODE Systems

In lieu of a computer algebra system, MathPlot can be used to create phase portraits for two- and three-dimensional ODE systems. Again, due to the low-resolution world, a transformation such as (1) must often be applied first. For example, the system

$$\frac{dx}{dt} = 2x - 3x^2 - 4xy, \quad \frac{dy}{dt} = -y + 3xy$$

exhibits a variety solution trajectories in the window $-\frac{1}{2} \leq x \leq 1$, $-\frac{1}{2} \leq y \leq 1$. But, in order for the trajectories to have sufficient detail in Minetest, the system should be scaled by a relatively large factor, e.g. 80. Applying (1) with $S_x = S_y = 80$ and $T_x = T_y = 0$, the differential equations become

$$\frac{dx}{dt} = 2x - \frac{3}{80}x^2 - \frac{1}{20}xy, \quad \frac{dy}{dt} = -y + \frac{3}{80}xy \quad \text{on } -40 \leq x, y \leq 80. \tag{3}$$

Then a phase portrait of this system can be constructed with the `ODE Plot` screen using a variety of initial data and $dz/dt = 0$, $z(0) = 0$. (See Figure 10.)

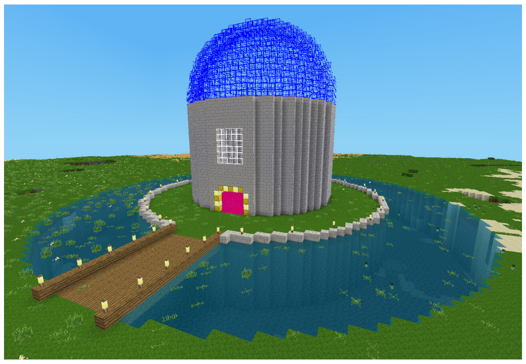
Indeed, the geometry of the game presents a need to apply scalings to equations, and this skill transfers directly to applications such as non-dimensionalization. Moreover, the other ploting capabilities of the MathPlot mod can be leveraged to explore further geometric aspects of the system. For example, the non-trivial nullclines of (3), i.e. the non-trivial curves along which $dx/dt = 0$ or $dy/dt = 0$, are (respectively)

$$2 - \frac{3}{80}x - \frac{1}{20}y = 0 \quad \text{and} \quad -1 + \frac{3}{80}x = 0,$$

and these curves can be plotted using the `Implicit Plot` screen. (They intersect at the equilibrium point in the first quadrant.)

Figure 11: Sonia Kovalevsky Math Day activity at Rose-Hulman.

Participation in the event. Photo credit: Rose-Hulman Institute of Technology / David Essex



The Observatory, built (mostly) using mathematical equations!

4.4 Extracurricular Activities / Projects

In addition to the occasional class activity in Calculus III, I have also used Minetest with the MathPlot mod for larger-scale activities and course projects.

4.4.1 Sonia Kovalevsky Math Day Activity

In February 2020, I conducted a Minetest activity for high school girls as part of the annual Sonia Kovalevsky Math Day at the Rose-Hulman Institute of Technology [6]. Mathematically, this activity focused on the Pythagorean Theorem, its equivalence to the "distance formula", and how it is used to set up equations for cylinders and spheres in $\mathbb{R}^3$. The ultimate goal of the activity was to use equations to build the cylindrical walls and hemispherical dome of an "observatory" (see Figure 11).

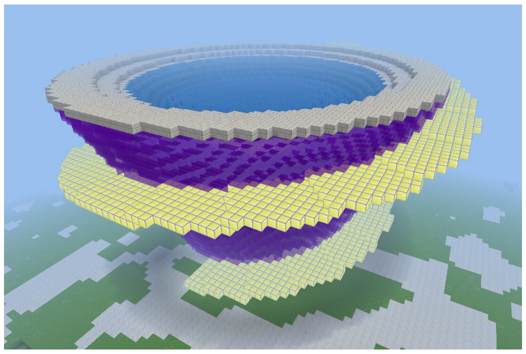
4.4.2 Vector Calculus Project

In my Spring 2020 Vector Calculus course, I assigned a project in which students were tasked with building an elevated swimming pool with a spiral staircase; see Figure 12a. In advance of assigning the project, I set up an internet-accessible Minetest server with the structure already built in it, and students could log in to explore what they were required to emulate and inspect the dimensions more carefully. As part of the project instructions, I also gave the students a cross-sectional schematic (see Figure 12b) to help identify the important dimensions before diving into finding equations, etc. For their final submissions, students uploaded the following to the LMS:

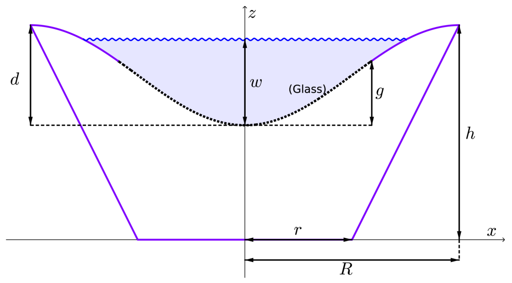
- A Maple file containing their prototyped equations and graphics.
- A zip file containing their Minetest world files so that I could verify and explore their creations.
- A typed, narrative report documenting their process of finding the relevant equations, prototyping the structure in Maple, and ultimately building it in the game.

In particular, the report was to include a graphic of the cross-sectional schematic with the dimensions labeled appropriately, a screenshot of their prototyped plots in Maple, a table

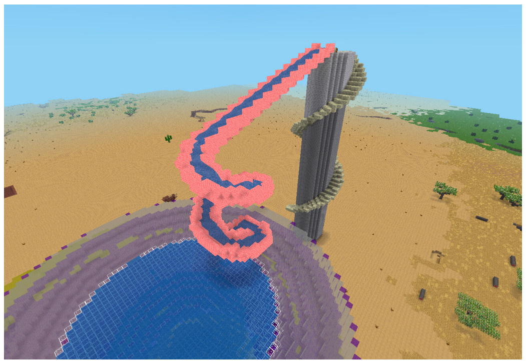
Figure 12: Vector Calculus Project



(a) Required structure for the project.



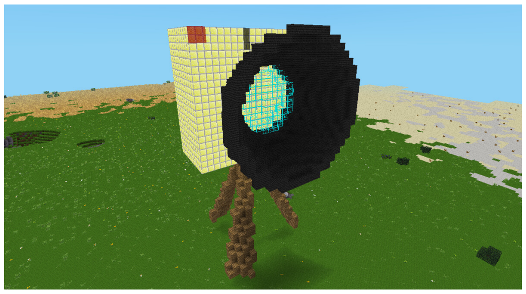
(b) Cross-sectional schematic of the structure.



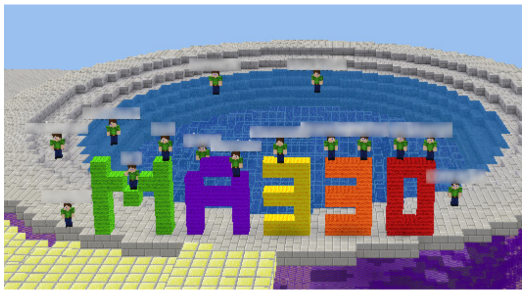
(c) One student built a waterslide into the swimming pool as their additional project item.

(d) Another student created a planet with rings and a multi-layered core for their additional project item.

Figure 13: Vector Calculus (MA330) Class Picture



(a) A student built a camera for the class picture using the MathPlot mod.



(b) Students joined for a group build and class picture (usernames redacted). This screenshot was taken while standing on top of the camera!

containing the parametric equations for each part of the structure and the relevant ranges of the parameters, and a screenshot of the finished structure in the game. Moreover, students were required to build an additional curve, surface, or solid of their own design (which they had proposed ahead of time for my approval), and I was very impressed by their creativity! See Figures 12c-12d for a couple of my personal favorite submissions.

As mentioned previously, this Vector Calculus class took place during the infamous Spring 2020 academic term, which was administered in an online modality due to the COVID-19 pandemic. It was *extremely* challenging for everyone (especially me!), but we ended this unprecedented and stressful term in a fun way–for the last day of class, we connected to the project Minetest server for a group build and "class picture" (see Figure 13). Having conducted this course mostly asynchronously, it felt good to leverage Minetest to create something together.

Acknowledgments

I would like to thank my colleague and friend William Green (green@rose-hulman.edu) for his helpful feedback and persistent encouragement to write this article.

Biographical Sketch

Kyle Claassen received his Ph.D. in mathematics from the Univeristy of Kansas and is currently an assistant professor at the Rose-Hulman Institute of Technology. He enjoys developing mathematical visualizations and is interested broadly in applied mathematics and computer programming.

REFERENCES

[1] Ahola, P., et al. (2022). *Minetest*, Version 5.6.0. https://www.minetest.net/

[2] Ahola, P., et al. (2023). Minetest Lua modding API reference. https://github.com/minetest/minetest/blob/master/doc/lua_api.txt

[3] Ierusalimschy, R. (2003). Programming in Lua (first edition). https://www.lua.org/pil/contents.html

[4] Claassen, K.M. (2021). *MathPlot (Minetest mod)*, Version 2.1.2. https://content.minetest.net/packages/kyleclaassen/mathplot/

[5] Claassen, K.M. (2021). MathPlot documentation. https://gitlab.com/kyleclaassen/minetest_mathplot/-/wikis/home

[6] Rose-Hulman Institute of Technology. (2020). Sonia math day to show high school girls the fun in math. https://www.rose-hulman.edu/news/2020/sonia-math-2020.html

[7] Ward, A. (2023). Minetest ContentDB. https://content.minetest.net/

[8] Ward, A. (2023). Minetest modding book. https://rubenwardy.com/minetest_modding_book/en/index.html

[9] Weisstein, E.W. (2003). Tanglecube (from MathWorld–a Wolfram web resource). https://mathworld.wolfram.com/Tanglecube.html

[10] Wikipedia. (2023). Lorenz system. https://en.wikipedia.org/wiki/Lorenz_system

[11] Wikipedia. (2023). Möbius strip. https://en.wikipedia.org/wiki/M%C3%B6bius_strip

[12] Wikipedia. (2023). Trefoil knot. https://en.wikipedia.org/wiki/Trefoil_knot