# Oriented and Non-Oriented Cubical Surfaces in the Penteract

Manuel Estévez[1], Érika Roldán[1,2], and Henry Segerman[3]

[1] ScaDS.AI, Leipzig University, Germany; estevez@informatik.uni-leipzig.de
[2] Max Planck Institute for Mathematics in the Sciences, Leipzig, Germany; roldan@mis.mpg.de
[3] Oklahoma State University, Stillwater, Oklahoma, USA; henry@segerman.org

## Abstract

Which surfaces can be realized with two-dimensional faces of the five-dimensional cube (the penteract)? How can we visualize them? In recent work, Aveni, Govc, and Roldán show that there exist 2690 connected closed cubical surfaces up to isomorphism in the 5-cube. They give a classification in terms of their genus $g$ for closed orientable cubical surfaces, and their demigenus $k$ for a closed non-orientable cubical surface. In this paper we present the definition of a cubical surface and we visualize the projection to $\mathbb{R}^3$ of a torus, a genus two torus, the projective plane, and the Klein bottle. We use reinforcement learning techniques to obtain configurations optimized for 3D-printing.

## Introduction

In Bridges 2023 [2], Estévez, Roldán, and Segerman presented various visualizations of 3D-printed representations of *closed connected orientable cubical surfaces* embedded in $\mathbb{R}^3$ on the two dimensional faces of the tesseract. These representatives are homeomorphic to either a sphere, a torus or two disconnected spheres; in fact, the maximum genus achievable for the tesseract is one. For a five-dimensional cube, often called a *penteract*, doing an exhaustive computational search, Aveni, Govc, and Roldán [1] found all orientable and non-orientable connected closed surfaces. Within these surfaces, by a result from Schulz [3], the maximum possible genus is five and the maximum possible demigenus is eight. They have also classified all closed surfaces in 2690 different isomorphism types.

Here, we have selected some of these surfaces to find good embeddings for visualizing them in $\mathbb{R}^3$ and for 3D-printing them: the genus 1 torus, the genus 2 torus, the projective plane, and the Klein Bottle. We work with non-orientable cubical surfaces, therefore we must deal with self intersections of its faces; in particular, we want to minimize them. After performing the perspective projection to 3D space, we assign a fix width (or radius) $w$ to all of its edges, resulting on cylindrical segments in 3D space. An edge overlap is a pair of cylindrical segments that intersect in 3D space. In order to have the best possible embedding for 3D printing and visualization of these surfaces, we implement a Reinforcement Learning (RL) algorithm that explores suitable three-dimensional embeddings which minimize self intersections of its faces and edge crossings for a fixed edge width $w$.

The rest of the paper is organized as follows. In Section , we introduce the notation and basic definitions of cubical surfaces in the penteract. In Section  we present some results and 3D-models of the configurations that the algorithm found for our selected surfaces, starting from particular initial configurations. In Section , we describe in detail the implementation of the RL algorithm.

## Cubical Surfaces in the Penteract

We denote the five-dimensional unit cube by $Q^5 = [0,1]^5$, and its set of vertices by $Q_0^5$. Each vertex of $Q^5$ can be represented by an element of the set of all five-tuples with binary entries $\{0,1\}^5$. We denote by $Q_1^5$ the one-dimensional skeleton of $Q^5$, that is, the set of its vertices $v$ and edges $e$. We observe that $Q_1^5$ is the graph

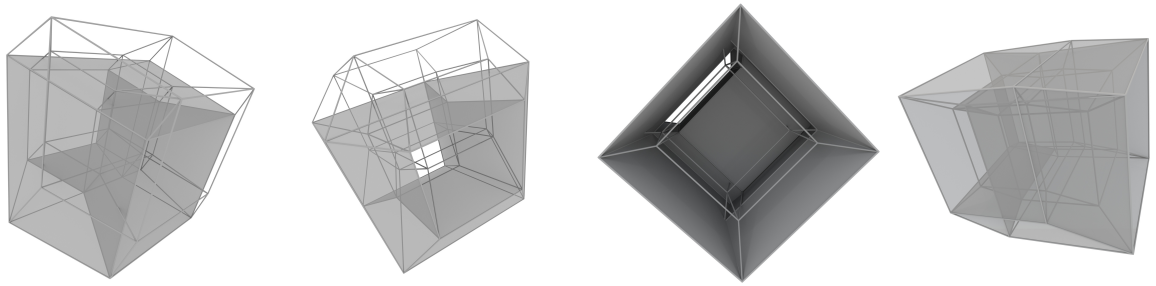
with vertex set $Q_0^5$ with an edge between two vertices if and only if they differ in exactly one coordinate. The cardinality of the set of faces $f$ containing an edge $e$ (resp. a vertex $v$) is denoted by $F_e$ (resp. $F_v$) and similarly the cardinality of the set of edges $e$ containing a vertex $v$ is denoted by $E_v$. Similarly, $Q_2^5$ denotes the two-dimensional skeleton of $Q^5$, which consists of the set of vertices $Q_0^5$, the one-dimensional skeleton $Q_1^5$, and all its two-dimensional faces $f$. We can continue this construction up to the penteract $Q_5^5$ itself, and name the elements of all the preceding sets the *cells* of $Q^5$. We refer to a subset of $Q_2^n$ as a *two-dimensional cubical complex*, which we will denote by $C$. We denote its set of vertices, edges and faces by $C_0$, $C_1$, and $C_2$ respectively. The *vertex figure* $\mathcal{F}_v$ of a vertex $v$ is the graph whose nodes are the edges in $C_1$ having $v$ as an endpoint and where two nodes $e, e' \in C_1$ are joined by an edge if and only if there is a face $f \in C_2$ with $e, e'$ as two of its edges. A *closed cubical surface* is a two-dimensional cubical complex $C$ in which every point has an open neighborhood homeomorphic to an open disk. This condition is equivalent to asking $C$ to fulfill the following conditions: Every edge is shared by exactly two faces, i.e. for all $e \in C_1$, $F_e = 2$, and the vertex figure $\mathcal{F}_v$ of any vertex $v \in C_0$ is a cyclic graph.

## Embeddings of Cubical Surfaces

For the following cubical surfaces we are using the results found by Aveni, Govc and Roldán [1], therefore we know that we are using the minimum number of faces needed for realizing these orientable and non-orientable surfaces.

### *Orientable Surfaces*

In Figure 1 (a) and (b) (resp. (c) and (d)) we present an embedding of a torus (resp. genus two torus) whose initial embedding had 19 (resp. 19) edge overlaps and 6 (resp. 33) face intersections. Our RL algorithm found embeddings with zero edge overlaps for both and zero (resp. 11) face intersections.

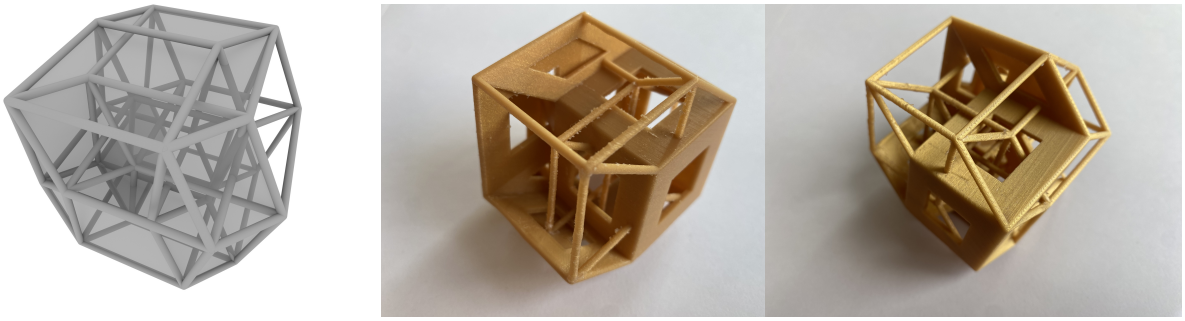


**(a)** *Genus 1 torus* **(b)** *Face intersections: 0* **(c)** *Genus 2 torus* **(d)** *Face intersections: 11*

**Figure 1:** *Orientable cubical surfaces. 3D models can be found in the supplementary files.*

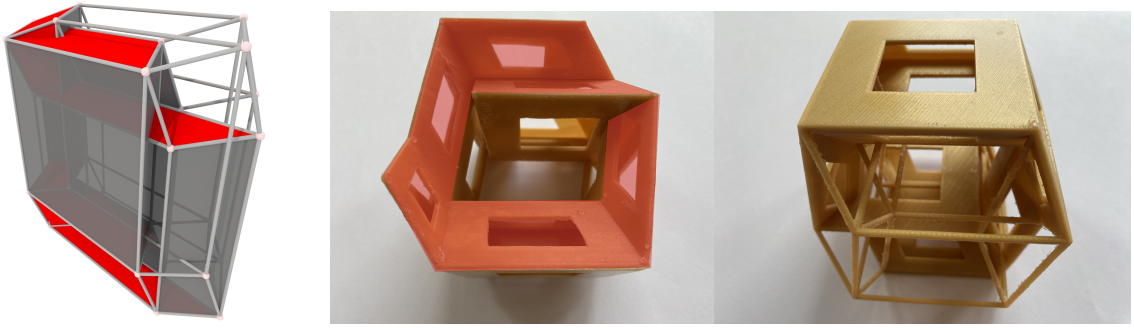
### *Non-Orientable Surfaces*

In Figure 2 (resp. Figure 3), we present an embedding of a cubical surface homeomorphic to the projective plane (resp. Klein Bottle), whose initial embedding had 19 (resp. 19) edge overlaps and 13 (resp. 9) face intersections. Our RL algorithm lowered the intersections to only three face intersections and zero edge overlaps for both of them. In Figure 3a, we give an embedding of the Klein Bottle with more than 3 intersections, emphasizing two Möbius strips that are subcomplexes of this surface. We invite the reader to count the number of intersections using our 3D model https://skfb.ly/oRB7H. Our projective plane and Klein bottle each have two cross-cap singularities. We are currently working on modifying our code to search for immersions without these singularities.



**(a)** *Face intersections: 3* **(b)** *Projective plane 3D print* **(c)** *Projective plane 3D print*

**Figure 2:** *A cubical surface homeomorphic to a Projective Plane. 3D model can be found in supplementary files.*



**(a)** *Face intersections* > 3. **(b)** *Klein Bottle & Möbius strip* **(c)** *Klein Bottle 3D print*

**Figure 3:** *A cubical surface homeomorphic to a Klein Bottle. 3D model can be found in supplementary files.*

## Using Reinforcement Learning to Minimize Face Intersections and Edge Overlaps

The embedding of the cubical complex $C$ in $\mathbb{R}^3$ is parameterized by a vector $s_t = (d_5, d_4, \phi_1, \cdots, \phi_{10}) \in \mathbb{R}^{12}$ at time $t$. We call $s_t$ a *state*. The value $d_5$ (resp. $d_4$) is the *distance from the camera point to the origin* for the perspective projection $pr_5(d_5) : \mathbb{R}^5 \to \mathbb{R}^4$ (resp. $pr_4(d_4) : \mathbb{R}^4 \to \mathbb{R}^3$) in five (resp. four)-dimensional space. When applying perspective projection in five (resp. four)-dimensional space, we fix the distance from the origin to the projection hyperplane to be 1 (resp. 10), and only vary $d_5$ and $d_4$. The values $\phi_i \in [0, 2\pi), (1 \leq i \leq 10)$ are the ten possible angles (one for each pair of axes) on which the penteract $Q^5$ and $C$ can be rotated around the origin in $\mathbb{R}^5$.

We call the set $\mathcal{S}$ of states, the *state space*, and denote the initial state by $s_0 \in \mathcal{S}$. The RL algorithm to which we refer simply as the *agent* can perform one of the possible actions in the *action set* $\mathcal{A} := \{\delta e_0, -\delta e_0, \delta e_1, -\delta e_1, \epsilon e_2, -\epsilon e_2, \cdots, \epsilon e_{11}, -\epsilon e_{11}\}$, where the $e_j \in \mathbb{R}^{12}$ are vectors whose $j$th coordinate is 1 and 0 elsewhere, and $\delta, \epsilon \in \mathbb{R}$ are positive real numbers defining the step length. After some testing we set these to $\delta = 0.5$ and $\epsilon = \pi/180$. An action $a \in \mathcal{A}$ is also a vector in $\mathbb{R}^{12}$. Given a state $s_t$ and an action $a$, we get the state $s_{t+1} = s_t + a$. The parameter $\delta$ affects the distances $d_5$ and $d_4$ while the parameter $\epsilon$ affects the rotation angles $\phi_i, (1 \leq i \leq 10)$. Therefore, an action can apply either a small five-dimensional rotation or move the five or four- dimensional cameras to obtain a better embedding. A *reward function* $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ gives feedback to the agent depending on whether the action taken favors a certain task. We explain the construction of our reward functions in Sections and for different tasks. The RL algorithm takes these reward functions and produces a probability distribution on the set of actions $\mathcal{A}$, conditional on

the current state. Given an initial state $s_0$ in $\mathcal{S}$, the RL algorithm applies a set of actions that take the agent to better and better states. For further details, see [4].

***Minimizing the number of intersecting faces***

For a pair of distinct faces $(f, f') \in \binom{C_2}{2}$, after performing some five-dimensional rotation, consider their perspective projections $pr(d_5, d_4, f)$ and $pr(d_5, d_4, f')$ in $\mathbb{R}^3$. If these projections intersect, they do so transversely (in a point or a line) or they overlap (we don't consider faces sharing an edge as intersecting). At the state $s_t \in \mathcal{S}$ the *number of face intersections* running through all pairs $(f, f')$ is denoted by $\Sigma_C(s_t)$. For most of our cubical surfaces we don't know the minimum number of face intersections, but we can propose a minimum $\Sigma_{C\text{prop}}$ and expect the algorithm to find a state $s_t$ such that $\Sigma_C(s_t) \leq \Sigma_{C\text{prop}}$. The reward function

$$R_1(s_t, a_t) := \begin{cases} 10 * (1 - \Sigma_C(s_t + a_t) + \Sigma_{C\text{prop}}) & \text{if } \Sigma_C(s_t + a_t) \leq \Sigma_{C\text{prop}} \\ \frac{\Sigma_C(s_t+a_t) - \Sigma_C(s_t)}{\Sigma_C(s_t)} & \text{if } \Sigma_C(s_t + a_t) > \Sigma_{C\text{prop}}, \end{cases}$$

will reward the agent when the action $a_t$ reduces $\Sigma_C(s_t + a_t)$ with respect to $\Sigma_C(s_t)$.

***Minimizing edge overlaps for 3D-printing***

For a pair of distinct edges $(e, e') \in \binom{C_1}{2}$, after performing some five-dimensional rotation, consider their perspective projections $pr(d_5, d_4, e)$ and $pr(d_5, d_4, e')$ in $\mathbb{R}^3$. To 3D-print the three-dimensional projection of the cubical surface we assign a constant width $r$ to all of the projected edges. Each pair of projected edges can overlap with each other at a given state $s_t \in \mathcal{S}$ if the perpendicular line segment $L_{e,e'}(s_t)$ connecting them has magnitude $|L_{e,e'}(s_t)| < 2r$. The *number of overlapping edges* at a state $s_t \in \mathcal{S}$ is denoted by $o_w(s_t)$. The reward function

$$R_2(s_t, a_t) := \begin{cases} 10 * (1 - \Sigma_C(s_t + a_t) + \Sigma_{C\text{prop}}) & \text{if } o(s_t) = 0 \text{ and } \Sigma_C(s_t + a_t) \leq \Sigma_{C\text{prop}} \\ \frac{o(s_t+a_t) - o(s_t)}{o(s_t)} & \text{if } o(s_t) \neq 0 \text{ and } \Sigma_C(s_t + a_t) \leq \Sigma_{C\text{prop}} \\ 0 & \text{if } \Sigma_C(s_t + a_t) > \Sigma_{C\text{prop}}, \end{cases}$$

will reward the algorithm when the action $a_t$ reduces $o_w(s_t + a_t)$ with respect to $o_w(s_0)$ if the number $\Sigma_{C\text{prop}}$ is achieved or improved. We prevent the agent from lowering $o(s_t)$ simply by decreasing the parameters $d_5, d_4$ because we want the width $r$ not to be too small with respect to the final size of the projection for 3D printing purposes. At a state $s_t \in \mathcal{S}$, we consider $L(s_t) := \sum_{(e,e') \in \binom{C_1}{2}} |L_{e,e'}(s_t)|$. The function $R_3(s_t, a_t) := \frac{L(s_t) - L(s_t + a_t)}{L(s_t)}$, will reward the agent for reducing $L(s_t)$. We take $R_4(s_t, a_t) := 1$ if $L(s_t) < \min(\{L(s_i)_{0 \leq i \leq t-1}\})$ and $R_4(s_t, a_t) := 0$ otherwise. We take then $R := R_1 + R_2 + R_3 + R_4$.

## References


[1] A. Aveni, D. Govc, and E. Roldán. "Cubical Surfaces." *(In preparation)*.

[2] M. Estévez, and E. Roldán, and H. Segerman. "Surfaces in the Tesseract." *Proceedings of Bridges 2023: Mathematics, Art, Music, Architecture, Culture*, Halifax, Canada, July 27 -31, 2023, pp. 441–450. http://archive.bridgesmathart.org/2023/bridges2023-441.html

[3] C. Von Schulz. "Geschlossene Flächen im Rand des Würfels." *Abh.Math.Semin.Univ.Hambg.*, Volume 50, 1980, pp. 89–94. https://doi.org/10.1007/BF02941416

[4] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Second Edition, 2018. http://incompleteideas.net/book/the-book-2nd.html