Convex Projective Surface Visualisation Tool

Sepehr Saryazdi, Supervised by Stephan Tillmann

February 18, 2022

Abstract

This report provides a practical software to visualise convex projective structures and compute algorithms which prove useful for research purposes. The software provides three variants of visualisation; cloverleaf position, hypersurface and projected hypersphere. The software also allows individuals to explore unique triangulations through an interface that can add custom triangles. In answering certain theoretical questions about the canonical cell decomposition (CCD), algorithms are implemented to compute the CCD and the $\mathcal{A}$-coordinates of their centres. The report finally provides a method of expressing $\mathcal{A}$-coordinates in a different basis such as to simplify the representation of desired outitude inequality conditions on the torus.

1 Introduction

Convex projective structures marry the study of algebraic topology and geometry through triangulations of orientable and punctured manifolds. Manifolds of this type, denoted $S := S_{g,n}$ with genus g and $n \geq 1$ punctures, have certain properties that are useful in physics and mathematics research. To study these surfaces, one can choose a triangulation $\Delta \subset S$ of the surface S and study the properties of Δ. These triangulations can then be given certain $\mathcal{A}$-coordinates, two positive numbers for each unique $e \in E$ and one positive number for each triangle $t \in T$. The moduli space of all possible coordinates for this triangulation is thus denoted by $\mathcal{A}_\Delta \cong \mathbb{R}_{>0}^{|T|+2|E|}$ for $E \subset \Delta$ edges and $T \subset \Delta$ triangles.

Once $x \in \mathcal{A}_\Delta$ is chosen, we can construct the fundamental domain Ω by gluing exactly one representative triangle from every triangle class in Δ. We then consider the moduli space of doubly-decorated structures $\mathcal{T}_f^\ddagger(S)$. It is known that one can choose certain $\mathbb{RP}^2$ coordinates for x as there exists a homeomorphism $\psi_\Delta : \mathcal{T}_f^\ddagger(S) \to \mathcal{A}_\Delta$ with $\psi_\Delta^{-1}(x)$ well defined (Tillmann et al., 2019). This choice is referred to as double-decoration as one chooses $c \in \mathbb{R}^3$ and $r \in (\mathbb{R}^3)^*$ satisfying certain properties for each vertex $v = e \cap e'$. These coordinates can then be visualised as a developing map $\tilde{\Omega} \subset \mathbb{RP}^2$ in a cloverleaf chart, $\mathbb{R}^3$ or a projection of $\mathbb{S}^3$. The Convex Projective Structure Visualisation Tool (CPSVis) was produced to handle these visualisations and perform multiple computations which will be presented in this report.

When such triangulations are constructed, there are many degrees of freedom for how one triangulates $S_{g,n}$ and how one gives $\mathcal{A}$-coordinates to Δ. For this reason, it is crucial to study the relationship between different triangulations Δ and Δ' and different coordinates $x, x' \in \mathcal{A}_\Delta$. A method of characterising an optimal triangulation Δ is to consider the canonical cell decomposition (CCD) of a doubly-decorated projective structure $x \in \mathcal{T}_f^\ddagger(S)$. The CCD relies on a function $\mathrm{Out}_x : \mathcal{A}_\Delta \to \mathbb{R}$. This outitude function satisfies the property that $\mathrm{Out}_x(e) \geq 0 \ \forall e \in E$ implies that x is convex in $\mathbb{R}^3$ and in the projection of $\mathbb{S}^3$ (Tillmann et al., 2019; Cooper and Long, 2015). This nice property forms the motivation to define the CCD of $x \in \mathcal{T}_f^\ddagger$ to be given by $\mathrm{CDD}(x)$ which satisfies $\mathrm{Out}_x(e) \geq 0 \ \forall e \in E \subset \mathrm{CCD}(x)$. From hereon, if $x \in \mathcal{T}_f^\ddagger(S)$ then $\mathrm{CCD}(\alpha) := \mathrm{CCD}(x)$ and $\mathrm{Out}_\alpha(e) := \mathrm{Out}_x(e)$ for $\alpha = \psi_\Delta(x) \in \mathcal{A}_\Delta$.

For any $\Delta \subset S$ which is a canonical cell decomposition, we define (Tillmann et al., 2019):

$$\mathring{\mathscr{C}}(\Delta) = \{x \in \mathcal{T}_f^\ddagger(S) \mid \mathrm{CCD}(x) = \Delta\}$$

$$\mathscr{C}(\Delta) = \{x \in \mathcal{T}_f^\ddagger(S) \mid \mathrm{CCD}(x) \subseteq \Delta\}$$

A natural way to obtain $\Delta = \mathrm{CCD}(x)$ for any Δ is to choose $x = \mathbb{1} := \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^T \in \mathcal{A}_\Delta$. $\mathbb{1}$ is called the centre of $\mathscr{C}(\Delta)$ and is given by $C(\Delta) := \mathbb{1} \in \mathcal{A}_\Delta$. Note that, relative to another triangulation Δ', the $\mathcal{A}$-coordinates of $C(\Delta)$ are no longer $\mathbb{1}$, and instead the coordinate $[\mathbb{1}]_{\Delta'} \in \mathcal{A}_{\Delta'}$ denotes $C(\Delta')$.

This gives rise to a simple question how one could compute $[C(\mathrm{CCD}(x))]_\Delta$, the centre of triangulation $\mathrm{CCD}(x)$ relative to Δ given some $x \in \mathcal{A}_\Delta$. This report will outline a method of achieving this in subsection 3.4.

One of the known important results is that there is an unbounded potential number of edge flips required to reach a canonical cell decomposition (Tillmann and Wong, 2015). However, it is not easy to come up with explicit examples $x \in \mathcal{A}_\Delta$ that achieve this. An extension to this report asks how one can easily choose $x_{\Delta'} \in \mathcal{A}_\Delta$ such that for some desired choice of triangulation $\Delta' \subset S$, we have $\mathrm{CCD}(x_{\Delta'}) = \Delta'$. A method that can help solve this problem, by expressing the outitude conditions in a nicer coordinate system, will be presented for the torus.

2 CPSVis User Interface and Functionality

To use CPSVis, a working version of Python 3.8 or later will need to be installed. To run the software, ensure that packages numpy, matplotlib and pandas are installed by running the command below.

```
python3 -m pip install numpy matplotlib pandas
```

Once these are correctly installed, the following command can be run in the folder 'cpsvis':

```
python3 -m main.py
```

A visual interface should pop up as shown in Figure 1. Here we can write any initial parameters and click 'Add Initial Triangle', which forms a triangle as per the instructions in subsection 3.2. An example of an added triangle is shown in Figure 2.

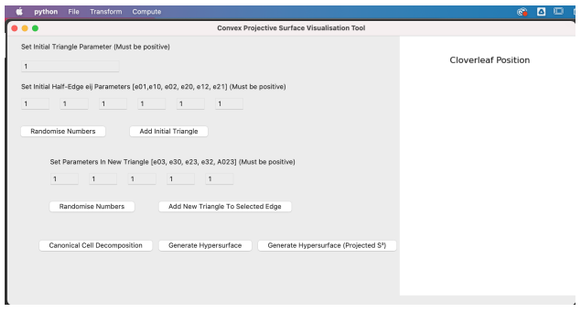

Figure 1: User Interface

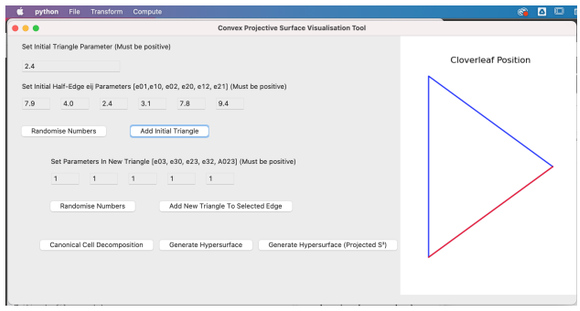

Figure 2: Added Initial Triangle

Using a mouse and clicking in the appropriate spot will make the software select an edge which is highlighted in red. An example of a different edge being selected in shown in Figure 3. Another triangle can be added by choosing appropriate parameters and clicking 'Add New Triangle To Selected Edge', which computes the new triangle as per the instructions in subsection 3.2. An example of this is shown in Figure 4. This process can be repeated manually to generate a structure by hand, which can be useful for understanding how the software works.

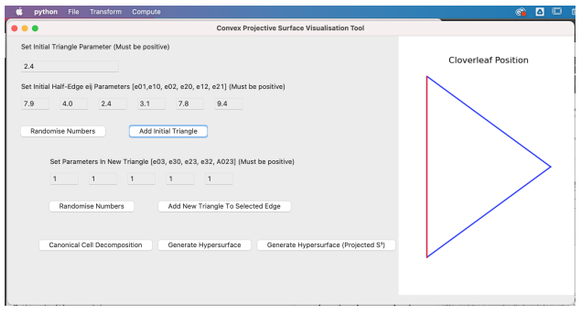

Figure 3: Selected Edge in Red

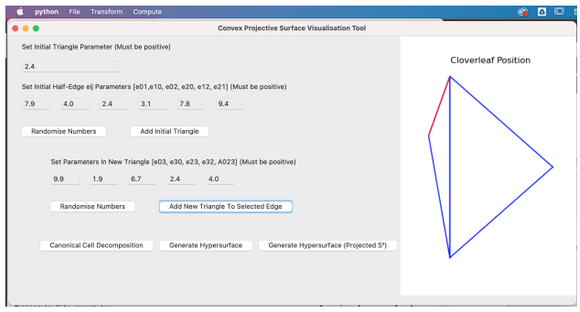

Figure 4: Added New Triangle

To automatically generate a developing map, a .csv file with the correct data structure formatting needs to be uploaded. There are two types of .csv triangulation data structures; gluing tables and parameter tables. The program accepts only gluing table formats as input and will assume all edge and triangle parameters are 1 by default. The program also accepts a combined gluing table and parameter table data structure. An example of these data structures are found in Table 1 and Table 2. To select a file to upload, click on the 'File' tab and then click on 'Import Gluing Table (CSV)' as shown in Figure 5. In this example, the '1-torus.csv' file from the 'example_gluing_tables' folder is selected and the program automatically visualises it as a combinatorial map as shown in Figure 5. Since this file is a pure gluing table data structure, then all parameters are assumed to be 1. In this interface, a triangle can be selected by clicking on it, which allows you to edit the edge and triangle parameter data as shown in Figure 6.

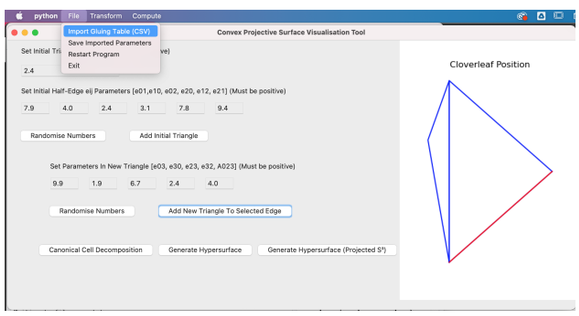

Figure 5: Import Gluing Table

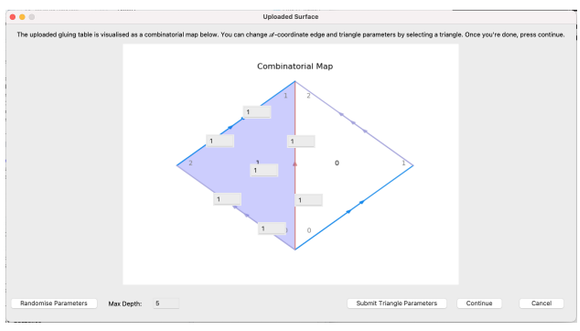

Figure 6: Edit Parameters of Selected Triangle

Once done entering triangle parameters, click 'Submit Triangle Parameters'. You may then select another triangle to edit, or click 'Continue'. When clicking continue, the program will automatically generate a cloverleaf chart similar to the picture in Figure 11. To view the $\mathbb{R}^3$ hypersurface or projected $\mathbb{S}^3$, click on 'Generate HYpersurface' or 'Generate Hypersurface (Projected $\mathbb{S}^3$)' respectively.

Other features such as saving imported gluing and parameter tables are also available by clicking 'File' and then 'Save Imported Parameters'. This data will be saved as a .csv file and can be re-uploaded later using the 'Import Gluing Table (CSV)' feature. To perform transformations on the imported data, use the 'Transform' tab. To compute information such as $[C(\mathrm{CCD}(x))]_\Delta$ as per subsection 3.4, use 'Compute' then click '$\mathcal{A}$-coordinates of Centre of Canonical Cell Decomposition' which will prompt to save the resultant gluing table and parameter files.

3 Theory and Methods

3.1 Gluing Tables and $\mathcal{A}$-coordinates

Gluing tables are data structures that store information about how each indexed triangle is glued together and their respective orientations. For the case of orientable surfaces, every triangle has a well-defined orientation. This orientation is conventionally chosen to be an anticlockwise numbering system for each vertex, with each edge defined to be oriented from a first number to a second number. An example of a triangulation of the torus is shown in Figure 7. Edge identification is denoted by parallel arrows and an equal number of repeated arrow heads. The corresponding gluing table for this triangulation is given by Table 1.

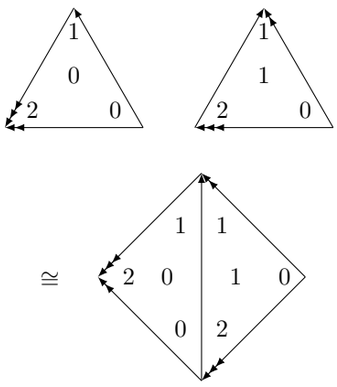

Figure 7: Triangle Constituents and Combinatorial Map of 1-Torus

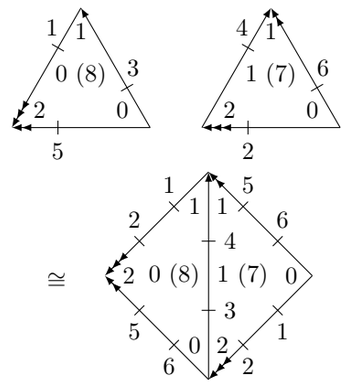

Figure 8: Assigned Parameters

Triangle	Edge 01	Edge 02	Edge 12
0	1 (21)	1 (01)	1 (02)
1	0 (02)	0 (12)	0 (10)

Table 1: Torus Gluing Table

Triangle Parameter	Edge 01	Edge 02	Edge 12
8	3	5	1
7	6	2	4

Table 2: Torus Parameter Table

To assign $\mathcal{A}$-coordinates to this triangulation, the convention is to assign the parameter to closer portion that begins the edge in an anticlockwise order. An example of this is shown in Figure 8, with corresponding parameter table shown in Table 2. To see this specific gluing table in the program, import the 1-torus-parameterised.csv file.

Once the coordinates have been designated and the gluing table has been laid out, one can organise it into the fundamental domain Ω, which contains exactly one representative of each triangle from the gluing table. An example of this is shown for the 2-torus in Figure 9, which is automatically visualised using CPSVis.

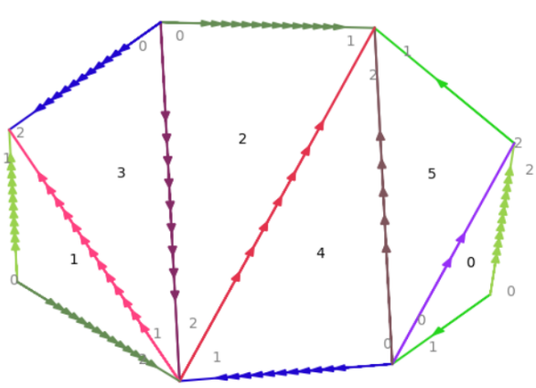

Figure 9: Combinatorial Map of 2-Torus

Using these triangulation and parameters, Ω can be extended into the developing map $\tilde{\Omega}$ by repeatedly adding triangles to the fundamental domain using the process outlined in subsection 3.2.

3.2 Computing $x \in \mathcal{T}_f^{\ddagger}(S)$ given $\alpha \in \mathcal{A}_\Delta$

To compute $x \in \mathcal{T}_f^{\ddagger}(S)$, we use the following procedure (Tillmann et al., 2019). Write $\mathcal{A}_\Delta \cong \mathbb{R}^{|T|+2|E|} = \mathbb{R}^{|T|} \times \mathbb{R}^{2|E|}$ and define $\mathcal{A}_{\Delta|T} = \mathbb{R}^{|T|}$ and $\mathcal{A}_{\Delta|E} = \mathbb{R}^{2|E|}$. We then write $\alpha = \begin{bmatrix} \alpha_{|T} \\ \alpha_{|E} \end{bmatrix}$ for $\alpha_{|T} \in \mathcal{A}_{\Delta|T}$ and $\alpha_{|E} \in \mathcal{A}_{\Delta|E}$. We choose some triangle parameter t, which is a component of $\alpha_{|T}$. We then start with a simple choice for vectors $C_0, C_1, C_2 \in \mathbb{R}^3$ such that they satisfy $\det(C_0|C_1|C_2) = t$. We then let e_{ij} be the edge parameters from $\alpha_{|E}$ corresponding to triangle t such that the ordering is anticlockwise in the gluing table with cyclic order $(e_{01}, e_{10}, e_{12}, e_{21}, e_{20}, e_{02})$. We then choose covectors $r_0, r_1, r_2 \in (\mathbb{R}^*)^3$ satisfying the following properties:

$$r_i \cdot C_j = e_{ij} > 0 \Leftrightarrow i \neq j$$
$$r_i \cdot C_j = 0 \Leftrightarrow i = j$$

Using this initial choice, another triangle with parameter t' can then be select according to the gluing table and the edge parameters $e_{03}, e_{30}, e_{23}, e_{32}$ are selected from t' with the orientation shown in Figure 10. The remaining details of how to compute r_3, C_3 are provided by Tillmann et al., (2019).

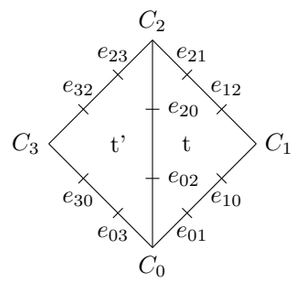

Figure 10: Doubly-decorated Initial Triangle

Repeating this process indefinitely, we obtain the developing map $\tilde{\Omega} \subset \mathbb{RP}^2$ which can be visualised in multiple ways as discussed in subsection 3.3.

3.3 Visualisations of $\tilde{\Omega}$

Upon obtaining $\tilde{\Omega} \subset \mathbb{RP}^2$, we can visualise it using three methods. This includes the cloverleaf chart, hypersurface and projected hypersphere. The cloverleaf chart allows one to easily identify the shape of the convex projective structure in $\mathbb{R}^2$, while the hypersurface and projected hypersphere are useful for visualising the lifted structure of $\tilde{\Omega}$ in $\mathbb{R}^3$.

3.3.1 Cloverleaf Chart

Since $\tilde{\Omega}$ has vectors in $\mathbb{R}^3$ assigned to every vertex which are all linearly independent by construction, it is possible to intersect these vectors to some affine plane $\mathcal{A} \cong \mathbb{R}^2 \subset \mathbb{R}^3$ by rescaling as we view these vectors to be representatives of elements in $\mathbb{RP}^2$. By choosing nice decorations for the initial triangle and choosing a nice affine plane $\mathcal{A}$, we can visualise $\tilde{\Omega}$ in a cloverleaf chart (Tillmann and Wong, 2015).

Let t be the $\mathcal{X}$-coordinate of the initial triangle parameter, which can be determined by the map $p_{\mathcal{X}} : \mathcal{A}_\Delta \to \mathcal{X}_\Delta$ (Tillmann et al., 2019). We then choose to make the initial vertices of the first triangle $[1:0:0]^T, [0:1:0]^T$ and $[0:0:1]^T$ with covectors $[0:\sqrt[3]{t}:1], [1:0:\sqrt[3]{t}]$ and $[\sqrt[3]{t}:1:0]$ respectively. Applying the same algorithm in subsection 3.2, we obtain $\tilde{\Omega}$ with these initial decorations. Let $\mathbf{b}$ be a basis defined by:

$$\mathbf{b} = \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\} = \left\{ \frac{1}{\sqrt{2}}(\mathbf{e}_2 - \mathbf{e}_1), \sqrt{\frac{2}{3}}\left(\mathbf{e}_3 - \frac{1}{2}(\mathbf{e}_1 + \mathbf{e}_2)\right), \frac{1}{3}(\mathbf{e}_1 + \mathbf{e}_2 + \mathbf{e}_3) \right\}$$

Let $\mathcal{A} = \{(x, y, z) \in \mathbb{R}^3 \mid x+y+z = 1\}$. It can be easily verified that the affine plane $\mathbf{b}_3 + \text{span}\{\mathbf{b}_1, \mathbf{b}_2\} = \mathcal{A}$. Construct the basis transformation matrix $T_{\mathbf{e},\mathbf{b}}$ given by:

$$T_{\mathbf{e},\mathbf{b}} = \begin{bmatrix} | & | & | \\ \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 \\ | & | & | \end{bmatrix} = \begin{bmatrix} -\frac{1}{\sqrt{2}} & -\frac{1}{2}\sqrt{\frac{2}{3}} & \frac{1}{3} \\ \frac{1}{\sqrt{2}} & -\frac{1}{2}\sqrt{\frac{2}{3}} & \frac{1}{3} \\ 0 & \sqrt{\frac{2}{3}} & \frac{1}{3} \end{bmatrix}$$

We note this has inverse given by:

$$T_{\mathbf{b},\mathbf{e}} = T_{\mathbf{e},\mathbf{b}}^{-1} = \begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{6}} & -\frac{1}{\sqrt{6}} & \sqrt{\frac{2}{3}} \\ 1 & 1 & 1 \end{bmatrix}$$

We then write any $\mathbf{v} \in \mathbb{RP}^2 = x\mathbf{e}_1 + y\mathbf{e}_2 + z\mathbf{e}_3 = [x : y : z]^T$. After transforming this vector, we then obtain new coordinates given by:

$$[\mathbf{v}]_{\mathbf{b}} = x'\mathbf{b}_1 + y'\mathbf{b}_2 + z'\mathbf{b}_3 = [x' : y' : z']^T = T_{\mathbf{b},\mathbf{e}}\mathbf{v}$$

We note that since $\text{span}\{\mathbf{b}_1, \mathbf{b}_2\}$ is the subspace that differs from a coset shift to the affine plane $\mathcal{A}$, then $\mathcal{A}$ can be viewed as a projective plane for $\text{span}\{\mathbf{b}_1, \mathbf{b}_2\}$ with standard coordinates given by:

$$[\mathbf{v}]_{\mathbb{R}^2} = \left(\frac{x'}{z'}, \frac{y'}{z'}\right)$$

Define $\alpha : \mathbb{R}^2 \to \mathbb{C}$ to be an isomorphism defined to satisfy:

$$\alpha([\mathbf{e}_1]_{\mathbb{R}^2}) = 1$$
$$\alpha([\mathbf{e}_2]_{\mathbb{R}^2}) = e^{\frac{2\pi}{3}i}$$

Where $[\mathbf{e}_1]_{\mathbb{R}^2}, [\mathbf{e}_2]_{\mathbb{R}^2}$ are the standard coordinates of $\mathbf{e}_1, \mathbf{e}_2 \in \mathbb{R}^3$ over $\mathcal{A}$ respectively.

After applying α onto every $[\mathbf{v}]_{\mathbb{R}^2} \in [\tilde{\Omega}]_{\mathbb{R}^2}$, we then obtain the cloverleaf chart. An example of a cloverleaf chart computed in CPSVis is shown in Figure 11.

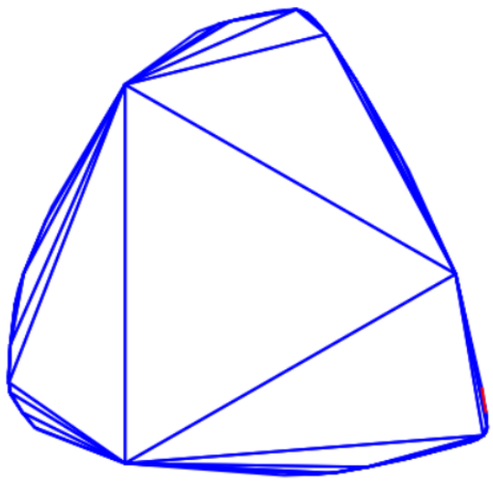

Figure 11: Cloverleaf Chart for some $x \in \mathcal{T}_f^{\ddagger}(S_{1,1})$

The nice thing about the cloverleaf chart is that the projective structure will be convex and is contained in the union of discs centred at the cube roots of -1 (Haraway and Tillmann, 2018).

3.3.2 $\mathbb{R}^3$ Visualisation

To visualise $\tilde{\Omega}$ in $\mathbb{R}^3$, we choose a certain set of initial decoractions to generate the developing map. Let t be the $\mathcal{A}$-coordinate triangle parameter and e_{ij} for t be the corresponding edge parameters as outlined in subsection 3.2. We then let $C_0 = \begin{bmatrix}\sqrt[3]{t} & 0 & 0\end{bmatrix}^T, C_1 = \begin{bmatrix}0 & \sqrt[3]{t} & 0\end{bmatrix}^T, C_2 = \begin{bmatrix}0 & 0 & \sqrt[3]{t}\end{bmatrix}^T$ with covectors $r_0 = \begin{bmatrix}0 & \frac{e_{01}}{\sqrt[3]{t}} & \frac{e_{02}}{\sqrt[3]{t}}\end{bmatrix}, r_1 = \begin{bmatrix}\frac{e_{10}}{\sqrt[3]{t}} & 0 & \frac{e_{12}}{\sqrt[3]{t}}\end{bmatrix}, r_2 = \begin{bmatrix}\frac{e_{20}}{\sqrt[3]{t}} & \frac{e_{21}}{\sqrt[3]{t}} & 0\end{bmatrix}$ respectively. Using the method in subsection 3.2, we then obtain $\tilde{\Omega}$. Since the vertices of $\tilde{\Omega}$ are vectors in $\mathbb{R}^3$, then this can be directly visualised as a hypersurface in $\mathbb{R}^3$.

An example of this visualisation computed in CPSVis is shown in Figure 12.

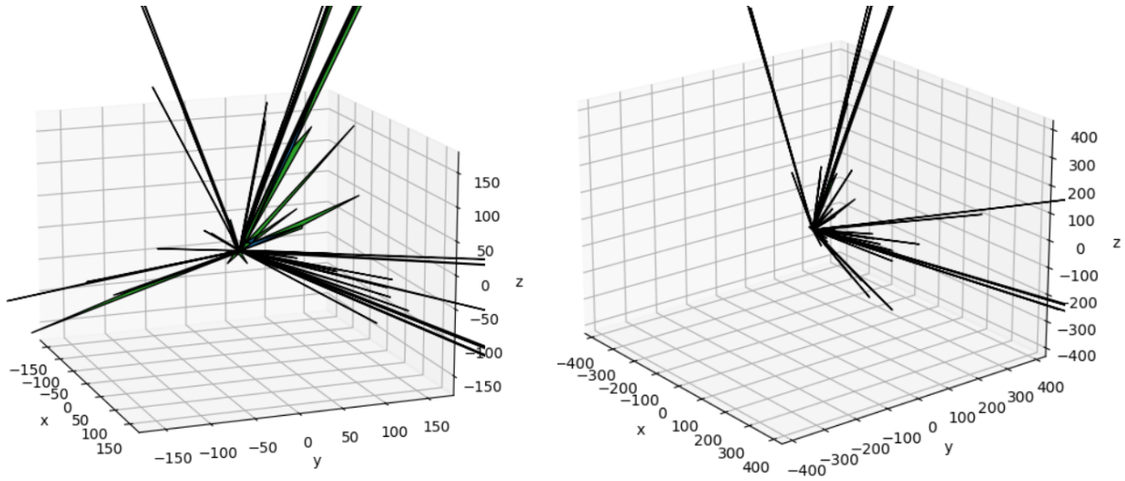

Figure 12: Hypersurface for some $x \in \mathcal{T}_f^{\ddagger}(S_{1,1})$

3.3.3 Projected $\mathbb{S}^3$ Visualisation

As demonstrated in Figure 12, it can be difficult to analyse the convexity of the projective structure as the magnitudes of decorations can become arbitrarily large. To handle this, one can instead view each vector $\mathbf{v} = \begin{bmatrix} x & y & z \end{bmatrix}^T \in \tilde{\Omega}$ as a representative element in $\mathbb{RP}^3$, given by $[\mathbf{v}]_{\mathbb{RP}^3} = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^T$. Since $\mathbb{RP}^3$ is homeomorphic to $\mathbb{S}^3$, then one can replace these representatives by $[\mathbf{v}]_{\mathbb{S}^3} = \begin{bmatrix} x' & y' & z' & w' \end{bmatrix}^T = \frac{1}{\sqrt{x^2+y+z^2+1}} \begin{bmatrix} x & y & z & 1 \end{bmatrix}^T$. We then project $\begin{bmatrix} x' & y' & z' & w' \end{bmatrix}^T \mapsto \frac{1}{1+x'} \begin{bmatrix} y' & z' & w' \end{bmatrix}^T \in \mathbb{R}^3$. This is then plotted directly as a hypersurface in $\mathbb{R}^3$.

An example of a projected $\mathbb{S}^3$ hypersurface computed in CPSVis is shown in Figure 13.

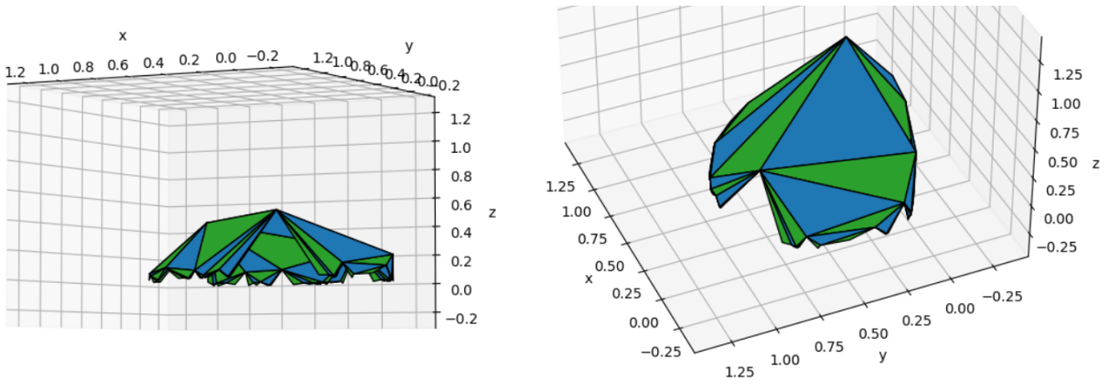

Figure 13: Projected $\mathbb{S}^3$ Hypersurface for some $x \in \mathcal{T}_f^{\ddagger}(S_{1,1})$

3.4 Computing $[C(\mathrm{CCD}(x))]_\Delta, x \in \mathcal{A}_\Delta$

To compute the centre of the canonical cell decomposition of $x \in \mathcal{A}_\Delta$ relative to $\mathcal{A}_\Delta$, we require the use of Theorem 3.1.

Theorem 3.1. CCD *is invertible with inverse* CCD^{-1}.

Proof. We note that the CCD algorithm is a sequence of edge flips (Tillmann and Wong, 2015) from Δ to $\mathrm{CCD}(x)$. Since flipping an edge and then flipping its resultant edge results in the identity, then each edge flip is invertible. This means that the CCD is a sequence of invertible operations, and thus CCD itself is invertible with inverse CCD^{-1}. □

To compute $[C(\mathrm{CCD}(x))]_\Delta$, we first begin by computing $\Delta' := \mathrm{CCD}(x)$. Using this triangulation, we then apply Theorem 3.1 to compute $\mathrm{CCD}^{-1}([C(\Delta')]_{\Delta'})$. Note that $[C(\Delta')]_{\Delta'} = \mathbb{1} \in \mathcal{A}_{\Delta'}$. We then note that the coordinates of any $[x]_{\Delta'} \in \mathcal{A}_{\Delta'}$ can be represented as $[x]_\Delta \in \mathcal{A}_\Delta$ by finding a sequence of edge flips between Δ' and Δ. Since CCD^{-1} forms a valid sequence of edge flips from Δ' to Δ, we thus have:

$$[C(\mathrm{CCD}(x))]_\Delta = [C(\Delta')]_\Delta = \mathrm{CCD}^{-1}([C(\Delta')]_{\Delta'})$$

This algorithm is available in CPSVis and allows the user to save the coordinates of $[C(\mathrm{CCD}(x))]_\Delta$ via a gluing and parameter csv file. An example of a computed centre for $S_{1,1}$ is shown in Figure 14.

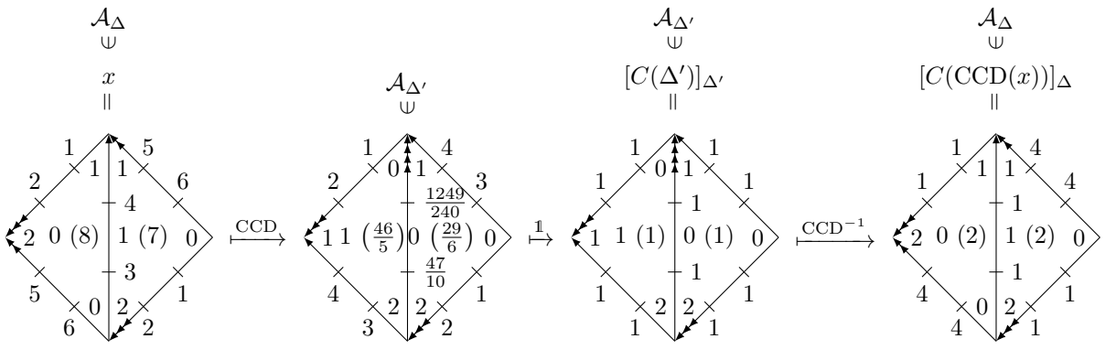

Figure 14: Centre of Canonical Cell Decomposition Algorithm for some $x \in \mathcal{A}_\Delta$

3.5 Extension: Re-expressing Outitudes in a Simpler Basis

One of the important results is that there is an unbounded potential number of edge flips required to reach a canonical cell decomposition (Tillmann and Wong, 2015). However, it is not easy to come up with explicit examples $x \in \mathcal{A}_\Delta$ that achieve this. This raises the following problem:

Given any Δ', compute an associated $x_{\Delta'} \in \mathcal{A}_\Delta$ such that $\mathrm{CCD}(x_{\Delta'}) = \Delta'$.

An algorithm to solve this problem for $S_{1,1}$ will be provided and some results that may be useful for completing the construction. The procedure is as follows:

Define $\tau_n := \Delta'$ and $\tau_0 := \Delta$. Construct a sequence of triangulations $[\tau_0, \tau_1, \tau_2, \cdots, \tau_{n-1}, \tau_n]$ and functions $g_k : \tau_k \to \tau_{k+1}$ such that $g := g_{n-1} \circ g_{n-2} \cdots \circ g_0$ is a function that maps τ_0 to τ_n. Furthermore, we require that $g_k \circ g_{k-1} \neq \mathbb{1}$ for each k, which is always possible by removing redundant edge flips.

We further denote e_k as the edge that was flipped by g_k and e'_k, e''_k as the other edges of triangulation k. A simple algorithm to ensure that for some $x \in \mathcal{A}_\Delta$ leads to $\mathrm{CCD}(x) = g(\tau_0)$ is by enforcing the following conditions denoted by Θ:

$$\begin{gathered}
\mathrm{Out}_{[x]_{\tau_k}}(e_k) < 0 \ \forall k \in \{0, \cdots, n-1\} \\
\mathrm{Out}_{[x]_{\tau_k}}(e'_k) \geq 0 \ \forall k \in \{0, \cdots, n-1\} \\
\mathrm{Out}_{[x]_{\tau_k}}(e''_k) \geq 0 \ \forall k \in \{0, \cdots, n-1\} \\
\mathrm{Out}_{[x]_{\tau_n}}(e_n) \geq 0 \\
\mathrm{Out}_{[x]_{\tau_n}}(e'_n) \geq 0 \\
\mathrm{Out}_{[x]_{\tau_n}}(e''_n) \geq 0
\end{gathered}$$

In order to enforce Θ, one needs to be able to systematically choose $[x]_{\tau_k}$ such that the outitudes have desired signs. For this change of basis, we shall assume $t = 1 \ \forall t \in T$. This means that for the torus with $|T| = 2$ and $|E| = 3$, $\mathcal{A}_\Delta \cong \mathcal{A}_{\Delta|T} \times \mathcal{A}_{\Delta|E}$ can be simplified to $(1,1) \times \mathcal{A}_{\Delta|E}$. We will let $x \in \mathcal{A}_{\Delta|E}$.

Write $x = \begin{bmatrix} a^- & a^+ & b^- & b^+ & e^- & e^+ \end{bmatrix}^T$ for edges a, b, e repsectively with entries as the edge parameters in an anticlockwise order as shown in Figure 15.

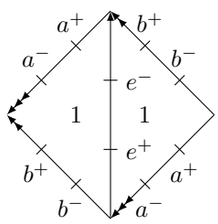

Figure 15: Arbitrary Parameter Coordinates on Torus

In this picture, the edge that is on the interior is e. To change coordinates so that the interior edge that is a or b, we define $\Lambda = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ and $S(\Lambda) = \begin{bmatrix} 0 & \Lambda & 0 \\ 0 & 0 & \Lambda \\ \Lambda & 0 & 0 \end{bmatrix}$ as a block matrix.

One can then verify that $S(\Lambda)x$ rearranges the coordinates such that a is the interior edge and $S(\Lambda)^{-1}x$ rearranges the coordinates such that b is the interior edge. This allows us to compute the outitudes for edges a, b, e:

$$\mathrm{Out}(e) = 1(e^+a^+ + e^-b^- - e^+e^-) + 1(e^+b^+ + e^-a^- - e^-e^+) = e^+(a^+ + b^+) + e^-(a^- + b^-) - 2e^-e^+$$

Using a similar process for a, b, we obtain:

$$\mathrm{Out}(a) = a^-(b^- + e^-) + a^+(b^+ + e^+) - 2a^-a^+$$

$$\mathrm{Out}(b) = b^-(e^- + a^-) + b^+(e^+ + a^+) - 2b^-b^+$$

We will write this in order a, b, e:

$$\begin{aligned}\text{Out}(a) &= a^-(b^- + e^-) + a^+(b^+ + e^+) - 2a^-a^+\\ \text{Out}(b) &= b^-(e^- + a^-) + b^+(e^+ + a^+) - 2b^-b^+\\ \text{Out}(e) &= e^+(a^+ + b^+) + e^-(a^- + b^-) - 2e^-e^+\end{aligned}$$

We have 4 cases unique up to reordering.

Case 1:

$$\begin{bmatrix}\text{Out}(a)\\ \text{Out}(b)\\ \text{Out}(e)\end{bmatrix} \geq \mathbf{0}$$

Case 2:

$$\begin{bmatrix}\text{Out}(a)\\ \text{Out}(b)\\ -\text{Out}(e)\end{bmatrix} > \mathbf{0}$$

Case 3:

$$\begin{bmatrix}\text{Out}(a)\\ -\text{Out}(b)\\ -\text{Out}(e)\end{bmatrix} > \mathbf{0}$$

Case 4:

$$\begin{bmatrix}\text{Out}(a)\\ \text{Out}(b)\\ \text{Out}(e)\end{bmatrix} < \mathbf{0}$$

This report will show how to compute the simpler basis for Case 1. A similar procedure is possible for all other cases.

$$\begin{bmatrix}\text{Out}(a)\\ \text{Out}(b)\\ \text{Out}(e)\end{bmatrix} \geq \mathbf{0} \Leftrightarrow \begin{matrix}a^-(b^- + e^-) + a^+(b^+ + e^+) \geq 2a^-a^+\\ b^-(e^- + a^-) + b^+(e^+ + a^+) \geq 2b^-b^+\\ e^+(a^+ + b^+) + e^-(a^- + b^-) \geq 2e^-e^+\end{matrix}$$

$$\Rightarrow \left\langle \begin{bmatrix}a^-\\ a^+\\ b^-\\ b^+\\ e^+\\ e^-\end{bmatrix}, \begin{bmatrix}b^- + e^-\\ b^+ + e^+\\ e^- + a^-\\ e^+ + a^+\\ a^+ + b^+\\ a^- + b^-\end{bmatrix} \right\rangle \geq \left\langle \begin{bmatrix}a^-\\ a^+\\ b^-\\ b^+\\ e^-\\ e^+\end{bmatrix}, \begin{bmatrix}a^+\\ a^-\\ b^+\\ b^-\\ e^+\\ e^-\end{bmatrix} \right\rangle$$

We note that $\begin{bmatrix}b^- + e^- & b^+ + e^+ & e^- + a^- & e^+ + a^+ & a^+ + b^+ & a^- + b^-\end{bmatrix}^T$ is a linear transformation of x, and thus we can write it as $P^L x$. We also note that $\begin{bmatrix}a^+ & a^- & b^+ & b^- & e^+ & e^-\end{bmatrix}^T$ is also a linear transformation of x, so it can be written as $P^R x$.

$$\Leftrightarrow \langle x, P^L x\rangle \geq \langle x, P^R x\rangle$$

$$\Leftrightarrow \langle x, (P^L - P^R)x\rangle \geq 0$$

Let $T = P^L - P^R$.

$$\Leftrightarrow \langle x, Tx\rangle \geq 0 \Leftrightarrow 2\langle x, Tx\rangle \geq 0 \Leftrightarrow \langle x, Tx\rangle + \langle x, Tx\rangle \geq 0$$

$$\Leftrightarrow \langle T^T x, x\rangle + \langle x, Tx\rangle \geq 0 \Leftrightarrow \langle x, T^T x\rangle + \langle x, Tx\rangle \geq 0 \Leftrightarrow \langle x, (T + T^T)x\rangle \geq 0$$

We note that $(T + T^T)^T = T^T + T = T + T^T$ and thus $T + T^T$ is an orthogonal matrix. This implies that $T + T^T$ has an orthonormal eigenbasis $\{\mathbf{u}_i\}$ with eigenvalues $\{\lambda_i\}$. Since this is a basis, then one can write $x = \sum c_i \mathbf{u}_i$ for appropriate coefficients c_i.

$$\Rightarrow \langle x, (T + T^T)x\rangle = \left\langle \sum c_i \mathbf{u}_i^T, (T + T^T)\left(\sum c_i \mathbf{u}_i\right)\right\rangle = \left\langle \sum c_i \mathbf{u}_i^T, \sum c_i \lambda_i \mathbf{u}_i\right\rangle = \sum c_i^2 \lambda_i$$

Let $\{\lambda_i^+\}$ and $\{\lambda_i^-\}$ be the positive and negative eigenvalues respectively with corresponding coefficients c_i^+ and c_i^-. Since we require $\langle x, (T + T^T)x\rangle \geq 0$, then we have:

$$\sum (c_i^+)^2 \lambda_i^+ \geq \sum (c_i^-)^2 \lambda_i^-$$

We note that these inequalities define a direct hyperbolic conic section in $2|E|$-dimensional space. To ensure that $x \in \mathbb{R}^{2|E|}_{>0}$, we also require that:

$$x = \sum c_i \mathbf{u}_i > \mathbf{0}$$

This representation leads us to a new coordinate system $\{c_i\}$ and the restricted set:

$$\left\{(c_1, \cdots, c_{2|E|}) \middle| \sum c_i \mathbf{u}_i > \mathbf{0} \text{ and } \sum c_i^2 \lambda_i \geq 0\right\}$$

Due to this simpler form, it may be easier to study this set and find a solution to the proposed problem.

4 References

1 Casella, A. (2019). Branched Cauchy–Riemann Structures on Once-Punctured Torus Bundles. *Bulletin Of The Australian Mathematical Society, 100*(1), 173-176. doi: 10.1017/s0004972719000455

2 Cooper, D., & D. Long, D. (2022). A generalization of the Epstein-Penner construction to projective manifolds, *143*(10), 4561-4569. Retrieved from http://web.math.ucsb.edu/ cooper/58.pdf

3 Haraway, R., Löwe, R., Tate, D., & Tillmann, S. (2019). On Moduli Spaces of Convex Projective Structures on Surfaces: Outitude and Cell-Decomposition in Fock-Goncharov Coordinates. *ArXiv*. Retrieved from https://arxiv.org/pdf/1911.04176.pdf

4 Haraway„ R., & Tillmann, S. (2017). Tessellating the Moduli Space of Strictly Convex Projective Structures on the Once-Punctured Torus. *Experimental Mathematics, 28*(3), 369-384. doi: 10.1080/10586458.2017.1409671

5 Tillmann, S., & Wong, S. (2015). An algorithm for the Euclidean cell decomposition of a cusped strictly convex projective surface. *ArXiv*. Retrieved from https://arxiv.org/abs/1512.01645