

COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

TRIANGULATION OF IMPLICIT SURFACE WITH
SINGULARITIES
MASTER'S THESIS

2021
BC. KRISTÍNA KORECOVÁ

COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

TRIANGULATION OF IMPLICIT SURFACE WITH
SINGULARITIES

MASTER'S THESIS

Study Programme: Computer Graphics and Geometry
Field of Study: Mathematics
Department: Department of Algebra and Geometry
Supervisor: doc. RNDr. Pavel Chalmovanský, PhD.

Bratislava, 2021
Bc. Kristína Korecová



Univerzita Komenského v Bratislavе
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Kristína Korecová
Študijný program: počítačová grafika a geometria (Jednoodborové štúdium,
magisterský II. st., denná forma)
Študijný odbor: matematika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: Triangulation of implicitly defined surfaces
Triangulácia implicitne definovaných plôch

Anotácia:

Ciel:

Literatúra:

**Kľúčové
slová:**

Vedúci: doc. RNDr. Pavel Chalmovianský, PhD.
Katedra: FMFI.KAG - Katedra algebry a geometrie
Vedúci katedry: doc. RNDr. Pavel Chalmovianský, PhD.

Spôsob sprístupnenia elektronickej verzie práce:
prípustná pre vlastnú VŠ

Dátum zadania: 19.10.2021

Dátum schválenia: 20.10.2021

prof. RNDr. Július Korbaš, CSc.
garant študijného programu

.....
študent

.....
vedúci práce

Acknowledgements: TODO Acknowledgements

Abstrakt

TODO Abstrakt po Slovensky

Kľúčové slová: TODO Kľúčové slová

Abstract

TODO Abstract in English

Keywords: TODO Keywords

Contents

| | |
|---|-----------|
| Introduction | 1 |
| 1 Theoretical background | 3 |
| 1.1 Implicit surfaces | 3 |
| 1.2 ADE singularities | 7 |
| 1.3 Non-isolated surface singularities | 9 |
| 1.4 Tringulation of regular implicit surfaces | 9 |
| 1.5 Data structures for triangulation algorithm | 10 |
| 1.6 CSG modelling for implicit surfaces | 14 |
| 2 Our contributions | 19 |
| 2.1 Triangulation adaptive to the local curvature | 19 |
| 2.2 Triangulation of ADE singularities | 20 |
| 2.3 Triangulation of non-isolated singularities of translation surfaces | 31 |
| 3 Results | 33 |
| 3.1 Quality criteria | 33 |
| 3.2 Comparison with TODO | 33 |
| 4 Future work | 35 |
| Conclusion | 37 |
| Appendix A | 41 |
| Appendix B | 43 |

List of Figures

| | | |
|------|--|----|
| 1.1 | Implicit surfaces with corresponding equations | 3 |
| 1.2 | Isolated and non-isolated singularity | 4 |
| 1.3 | Normal cut | 5 |
| 1.4 | Visualisation of the curvature of the double-torus | 6 |
| 1.5 | TODO | 10 |
| 1.6 | Example of winged edge representation | 11 |
| 1.7 | Visualisation of the half-edge data structure | 12 |
| 1.8 | Example of half-edge representation | 12 |
| 1.9 | Example of CSG tree | 15 |
| 1.10 | Boolean operations on implicit curves | 16 |
| 1.11 | Intersection of a sphere and a plane | 17 |
| 2.1 | Adaptive height of the new triangle | 19 |
| 2.2 | A_{n--} singularities | 21 |
| 2.3 | A_{n+-} singularities | 21 |
| 2.4 | D_{n+-} singularities | 22 |
| 2.5 | D_{n--} singularities | 22 |
| 2.6 | Intersection of D_{n--} singularities with plane $z = 0$ | 23 |
| 2.7 | Triangulation vectors for two branches of D_{n--} singularities. | 23 |
| 2.8 | E_n singularities. | 23 |
| 2.9 | Intersection of E_{7++} and E_{8++} singularities with plane $z = 0$ | 24 |
| 2.10 | Triangulation of A_{n--} singularity. | 25 |
| 2.11 | Equidistant points on ellipse. | 25 |
| 2.12 | TODO. | 26 |
| 2.13 | Plane with singularities. | 28 |
| 2.14 | C^1 cosine bump function. | 29 |
| 2.15 | Construction of the cosine bump function using CSG | 30 |
| 2.16 | Attaching the singularity to a plane using the cosine bump function . . | 31 |
| 2.17 | Attaching the singularity to a plane using CSG | 32 |
| 2.18 | Plane with multiple attached singularities | 32 |

| | |
|--|----|
| 4.1 Attaching the singularity to a plane using CSG | 42 |
|--|----|

List of Tables

| | | |
|-----|--|----|
| 1.1 | Edge table of a winged edge data structure for the figure 1.6. | 11 |
| 1.2 | Vertex table of a winged edge data structure for the figure 1.6. | 11 |
| 1.3 | Face table of a winged edge data structure for the figure 1.6. | 11 |
| 1.4 | Edge table of a half-edge data structure for the figure 1.8. | 13 |
| 1.5 | Vertex table of a half-edge data structure for the figure 1.8. | 13 |
| 1.6 | Face table of a half-edge data structure for the figure 1.8. | 13 |
| 2.1 | Implicit equations for bump function modeling | 29 |

Introduction

In mathematics, implicit surface is a set of points where the implicit function $F : \mathbb{R}^3 \rightarrow \mathbb{R}$ is zero. Thanks to unification of constructive solid geometry modelling and implicit surfaces, we can now model and represent very complex surfaces using a single function.

In computer graphics, triangular mesh is the most common way of surface representation and visualization. Calculating the intersection of a ray and a triangular mesh is far easier task as calculating the intersection of a ray and an implicit surface.

In this thesis we follow up on our effort in bachelor's thesis [10] to present an algorithm for creating the triangular mesh of regular implicit surfaces.

First, we reimplement the algorithm to be more effective, by using advanced data structures for mesh representation and 3D point search.

Next, we extend this algorithm to include the triangulation of certain types of isolated singularities - ADE singularities. We analyze the geometry of these singularities and propose an approach to triangulate these singularities.

Lastly, we improve the adaptive technique, which changes the size of the triangles based on the local curvature of the surface.

Chapter 1

Theoretical background

1.1 Implicit surfaces

Implicit functions are a tool for surface representation and manipulation. In computer graphics, they can be used for modelling of complex surfaces using boolean operations, realistic animations, rendering and other.

Implicit functions do not define the boundary explicitly, instead the surface is defined as a zero set of a function.

Definition 1 Given a function $F : \mathbb{R}^3 \rightarrow \mathbb{R}$, one can define an implicit surface as a set of points that fullfil $F(x, y, z) = 0$.

Some examples of implicit surfaces and their equations can be seen on image 1.1.

Unit normal vector of the implicit surface in point (x_0, y_0, z_0) is normalized gradient of the implicit function in that point.

Definition 2 Gradient vector of an implicit function $F : \mathbb{R}^3 \rightarrow \mathbb{R}$ is defined as

$$\nabla F(x, y, z) = \left(\frac{\partial F(x, y, z)}{\partial x}, \frac{\partial F(x, y, z)}{\partial y}, \frac{\partial F(x, y, z)}{\partial z} \right).$$

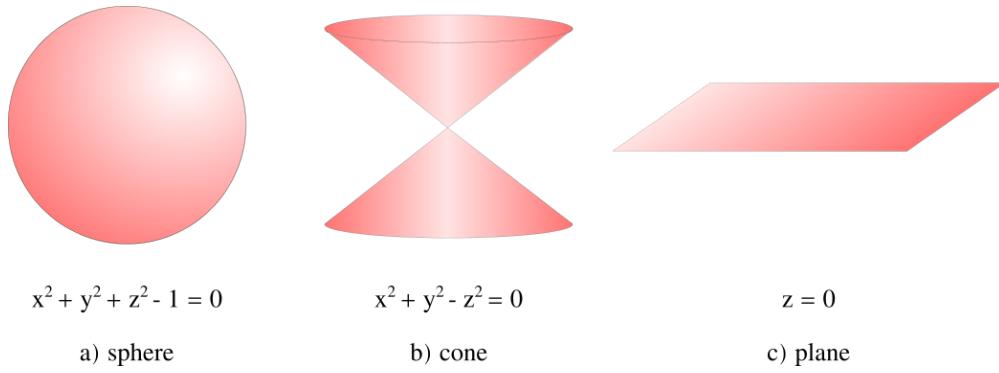


Figure 1.1: Implicit surfaces with corresponding equations.

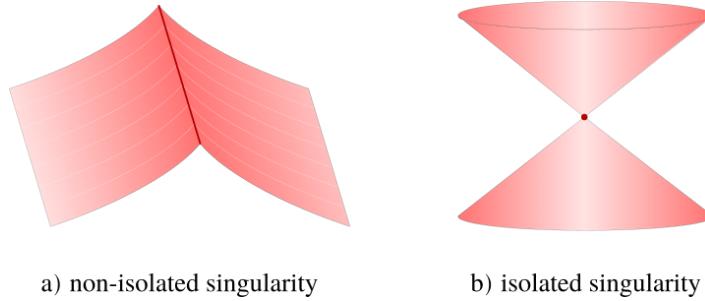


Figure 1.2: Isolated and non-isolated singularity.

If $\nabla F(x, y, z) \neq 0$, we can define the unit normal vector of F as a normalized gradient vector

$$N(F(x, y, z)) = \frac{\nabla F(x, y, z)}{\|\nabla F(x, y, z)\|} \quad \text{for } \nabla F(x, y, z) \neq 0.$$

Points lying on the implicit surface can be classified as regular or singular based on the value of the gradient vector in that point.

Definition 3 Point $P = (x, y, z)$ lying on the implicit surface is said to be regular, if $\nabla F(x, y, z) \neq 0$. On the contrary, point P is said to be singular, if $\nabla F(x, y, z) = 0$.

Singular points can be further classified as isolated or non-isolated based on their surroundings.

Definition 4 Singular point P is said to be isolated, if there exists an open ball $B_\varepsilon(P)$, which does not contain any other singular point. Singular point P is said to be non-isolated if it is not isolated.

On the image 1.2 we can see example of isolated and non-isolated singularities.

Curvature of a surface

Curvature is a fundamental concept in differential geometry of curves and surfaces. In case of curves, curvature is a measure of how much does the curve differ from a straight line. It is defined as the inverse of the radius of the osculating circle, which is the second order approximation of the curve.

For surfaces, curvature is a measure of how much does the surface differ from a plane. The definition of the curvature of a surface is not as straightforward as in the case of curves, as the curvature depends on the choice of the direction in which we measure the curvature.

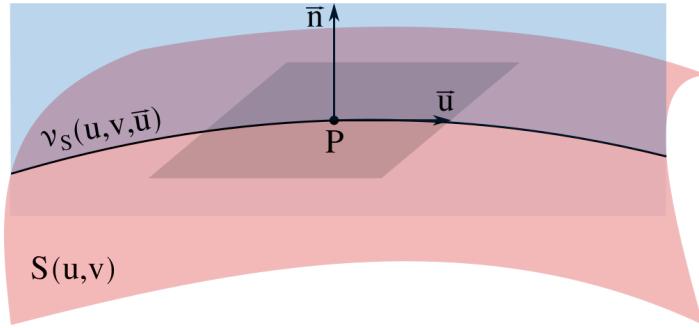


Figure 1.3: Normal cut of the parametric surface $S(u, v)$.

The idea of measuring the curvature of a surface has a long history in mathematics. One of the first contributors was a mathematician Carl Friedrich Gauss, who developed the idea of the Gaussian curvature of surfaces. In this subsection, we are drawing from the summary presented by Tiago Novello et al.[13].

Normal curvature of the surface

Let S be a parametric surface

$$S(u, v) : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

$$S(u, v) = (S_x(u, v), S_y(u, v), S_z(u, v)).$$

Let us denote the unit normal vector of the surface S in the point $S(u, v)$ as $\overrightarrow{n(u, v)}$.

We define the normal curvature of the surface as a function of the location of the point on the surface given by parameters u and v and the unit tangent vector in that point \overrightarrow{u} .

Definition 5 *Normal cut of a surface S in the regular point P in the direction of the unit tangent vector \overrightarrow{u} is defined as an intersection of the surface S and a plane given by the vectors \overrightarrow{u} and $\overrightarrow{n(u, v)}$.*

The visualisation of the normal cut is shown on the image 1.3. It is clear, that the normal cut is a plane curve lying on the surface, we denote this normal cut as $\nu_S(u, v, \overrightarrow{u})$.

Definition 6 *Oriented normal curvature of the surface in the regular point P in the direction of the unit tangent vector \overrightarrow{u} is defined as the curvature of the normal cut $\nu_S(u, v, \overrightarrow{u})$. Non-oriented normal curvature is defined as an absolute value of the oriented normal curvature.*

Definition 7 *Minimal and maximal curvature in the point $P = S(u, v)$ are defined as*

$$\kappa_{\min}(u, v) = \min_{\overrightarrow{u} \in T_P(u, v)} \nu_S(u, v, \overrightarrow{u}),$$

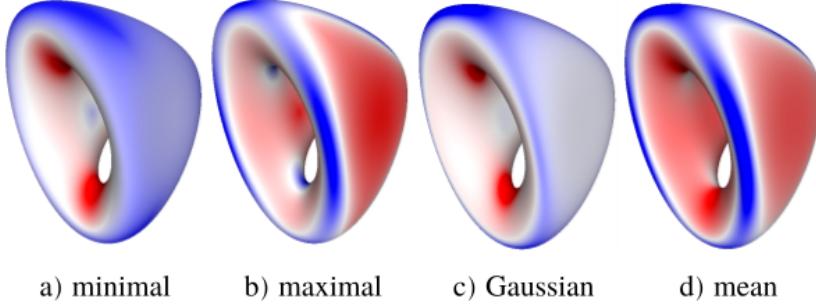


Figure 1.4: Visualisation of the curvature of the double-torus.

$$\kappa_{max}(u, v) = \max_{\vec{u} \in T_P(u, v)} \nu_S(u, v, \vec{u}),$$

where $T_S(u, v)$ is a tangent plane of the surface S in the point P .

Minimal and maximal curvature are called principal curvatures.

Definition 8 Gaussian curvature is defined as a product of principal curvatures:

$$\kappa_G(u, v) = \kappa_{min}(u, v)\kappa_{max}(u, v).$$

Gaussian curvature describes the shape of the surface in the local neighborhood of the point. The points where Gaussian curvature is positive are called elliptic points. The points where Gaussian curvature is negative are called hyperbolic points. The points where only one of $\kappa_{min}, \kappa_{max}$ is zero are called parabolic and the points where both κ_{min} and κ_{max} are zero are called planar. The shape of the surface in the local neighborhoods of the points is as follows:

- elliptic points → surface is curved like a sphere,
- hyperbolic points → surface is curved like a saddle,
- parabolic points → surface is curved like a parabolic cylinder,
- planar points → surface is flat.

Gaussian curvature is an intrinsic property, which means that it is also independent of the placement of the surface in the space.

Definition 9 Mean curvature is defined as an arithmetic mean of principal curvatures:

$$\kappa_M(u, v) = \frac{\kappa_{min}(u, v) + \kappa_{max}(u, v)}{2}.$$

Minimal, maximal, Gaussian and mean curvature are visualized on the image 1.4.

Curvature formulas for implicit surface

A version of curvature formulas for implicit surfaces appeared in [16] and were reformulated, summarized and prooved by Ron Goldman [6]. In this subsection we point out these formulas.

Let $F : \mathbb{R}^3 \rightarrow R$ be an implicit function which defines surface by the equation $F(x, y, z) = 0$. Let us denote $F_t = \frac{\partial F}{\partial t}$ and $F_{ts} = \frac{\partial^2 F}{\partial t \partial s}$. Hessian matrix - the matrix of second derivatives is defined as

$$H(F) = \begin{pmatrix} F_{xx} & F_{xy} & F_{xz} \\ F_{yx} & F_{yy} & F_{yz} \\ F_{zx} & F_{zy} & F_{zz} \end{pmatrix},$$

and the adjoint of the Hessian is defined as

$$H^*(F) = \begin{pmatrix} F_{yy}F_{zz} - F_{yz}F_{zy} & F_{yz}F_{zx} - F_{yx}F_{zz} & F_{yx}F_{zy} - F_{yy}F_{zx} \\ F_{xz}F_{zy} - F_{xy}F_{zz} & F_{xx}F_{zz} - F_{xz}F_{zx} & F_{xy}F_{zx} - F_{xx}F_{zy} \\ F_{xy}F_{yz} - F_{yx}F_{zy} & F_{yx}F_{xz} - F_{xx}F_{yz} & F_{xx}F_{yy} - F_{xy}F_{yx} \end{pmatrix}.$$

We can now formulate the formulas of Gaussian, mean, minimal and maximal curvature.

Gaussian curvature of the implicit surface defined by function F is given by

$$\kappa_G = \frac{\nabla F * H^*(F) * \nabla F^T}{|\nabla F|^4}.$$

Mean curvature of the implicit surface defined by function F is given by

$$\kappa_M = \frac{\nabla F * H^*(F) * \nabla F^T - |\nabla F|^2 \text{Trace}(H)}{2|\nabla F|^3}.$$

The principal curvatures κ_{min} and κ_{max} can be calculated from Gaussian curvature and mean curvature as

$$\kappa_{min}, \kappa_{max} = \kappa_M \pm \sqrt{\kappa_M^2 - \kappa_G}.$$

1.2 ADE singularities

ADE singularities, also reffered to as du Val singularities are a specific class of simple, isolated surface singularities. They were classified by Arnold's [1] according to ADE classification [7] based on correspondence of these singualrities to simply laced Dynkin diagrams [4]. We know infinitely many A singularities – A_1, A_2, \dots , infinitely many D singularities – D_4, D_5, \dots and three E singularities – E_6, E_7 and E_8 . ADE singularities are specified by their normal forms. When working in complex space, each singularity has a single normal form:

- $A_n \quad F(x, y, z) = x^{n+1} + y^2 + z^2,$
- $D_n \quad F(x, y, z) = yx^2 + y^{n-1} + z^2,$
- $E_6 \quad F(x, y, z) = x^3 + y^4 + z^2,$
- $E_7 \quad F(x, y, z) = x^3 + xy^3 + z^2,$
- $E_8 \quad F(x, y, z) = x^3 + y^5 + z^2.$

Each ADE singularity on a surface can be locally expressed by their normal form.

In the real case, changing the signs in these equations produces different surfaces and therefore, ADE singularities can be further classified by their signature.

Definition 10 Let's mark real surface singularities based on their signature as follows:

- $A_{n\pm\pm} \quad F(x, y, z) = x^{n+1} \pm y^2 \pm z^2,$
- $D_{n\pm\pm} \quad F(x, y, z) = yx^2 \pm y^{n-1} \pm z^2,$
- $E_{6\pm\pm} \quad F(x, y, z) = x^3 \pm y^4 \pm z^2,$
- $E_{7\pm\pm} \quad F(x, y, z) = x^3 \pm xy^3 \pm z^2,$
- $E_{8\pm\pm} \quad F(x, y, z) = x^3 \pm y^5 \pm z^2.$

The most common example of a surface with ADE singularity is an ordinary cone. Given as the zero set of the function $F(x, y, z) = x^2 - y^2 - z^2$, cone has a singular point $P = (0, 0, 0)$. This singular point is an example of A_{1--} singularity.

Correspondence between $SO(3, \mathbb{R})$ group and ADE singularities

$SO(3, \mathbb{R})$ is special orthogonal group over the field of real numbers in three dimensions. It is also called 3D rotation group, as it is a group of all rotations about the origin in \mathbb{R}^3 .

Definition 11 $SO(3, \mathbb{R})$ is a group of 3×3 orthogonal matrices of real numbers with determinant 1.

$$SO(3, \mathbb{R}) = \left\{ A \in \mathbb{R}^{3 \times 3} \mid AA^T = I, \det(A) = 1 \right\}.$$

Simply laced Dynkin diagrams correspond to all finite subgroups of $SO(3, \mathbb{R})$. Finite subgroups of $SO(3, \mathbb{R})$ are the rotational symmetry groups of

- pyramid with n vertices (cyclic subgroup \overline{C}_n),
- double pyramid with n vertices (dihedral subgroup \overline{D}_n),

- platonic solids
 - tetrahedron (tetrahedral subgroup \bar{T})
 - octahedron (octahedral subgroup \bar{O})
 - icosahehedron (icosahedral subgroup \bar{I})

The correspondence is as follows:

- $A_n \iff \bar{C}_{n+1}$,
- $D_n \iff \bar{D}_{n+2}$,
- $E_6 \iff \bar{T}$,
- $E_7 \iff \bar{O}$,
- $E_8 \iff \bar{I}$.

The conclusion is that ADE singularities correspond to finite subgroups of $SO(3, \mathbb{R})$, which represent certain types of symmetries in \mathbb{R}^3 .

1.3 Non-isolated surface singularities

1.4 Tringulation of regular implicit surfaces

In this section, we shortly describe the algorithm for triangulation of the regular parts of implicit surfaces introduced in bachelor's thesis [10].

The algorithm creates triangles iteratively, one at a time. The new triangle is created at the edge of the existing triangle. The new point is projected on the surface using the Newton-Rhapson method. This method finds the root of the implicit function lying on the line, which is in the direction of the gradient of the implicit function. This approach is displayed on the figure 1.5.

After the new point is projected on the surface, conditions are checked. Some of these conditions are based on the Delaunay triangulation introduced by Hilton [8]. The Delaunay condition checks, if some other points are in the proximity of circumcenter of the new triangle. The conditions minimize the chances of triangle intersection. If the new triangle intersects with some existing triangles, the algorithm tries to connect the new triangle to the existing triangles in its proximity.

The algorithm is enriched with the possibility to triangulate adaptively to the curvature of the surface using approach presented by Akkouche [2]. It can also triangulate surfaces in the bounded volume - axis aligned bounding box, given by six numbers - minimal and maximal value for each of the three axes.

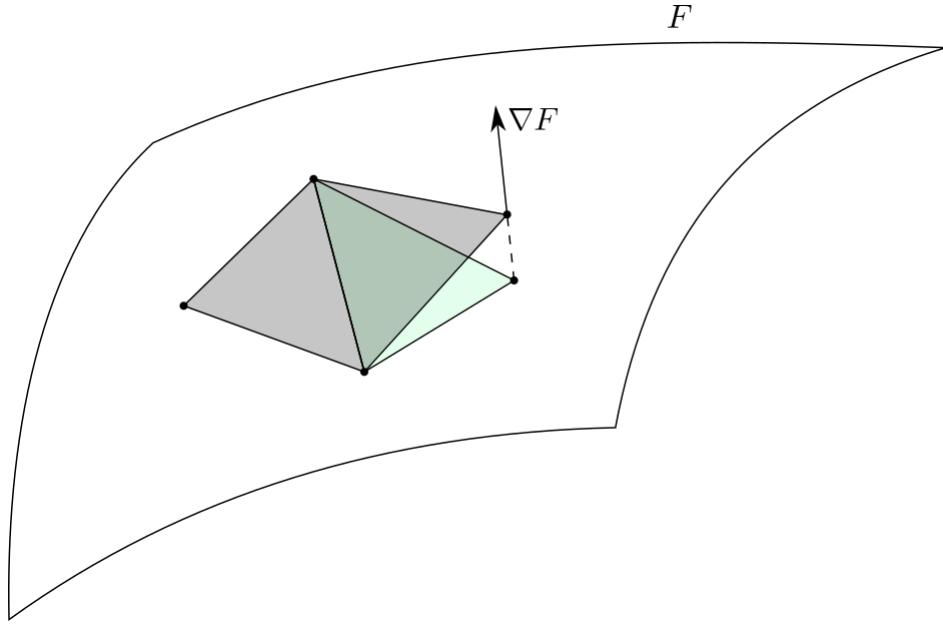


Figure 1.5: TODO

As a part of our work, we reimplement the algorithm to be more effective by using advanced data structures, such as the half-edge data structure [9] and the range tree [11].

1.5 Data structures for triangulation algorithm

Half-edge data structure

Triangular mesh is given by a set of vertices, non-oriented edges and triangular faces. There are multiple methods for mesh representation. The straightforward one - list of vertices, edges and faces does not provide any information about the local surroundings of the vertices, edges and faces and therefore the searching for incident faces or incident edges is complicated and inefficient.

In 1975, Baumgart [3] presented a representation using winged edges, which was further improved in 1985 by Weiler [17] who presented the modification called half-edge data structure. Both of these representations are edge-based representations, each edge stores references (pointers) to the surrounding vertices, edges and faces. One can easily extract the information of the surrounding vertices, edges and faces.

In the winged edge representation, each edge is represented as oriented line segment and stores references to its vertices, both incident faces, left and right traverse. Each vertex stores a reference to one of the outgoing edges and each face stores a reference to one of its incident edge. An example of such representation is shown on the figure 1.6 and tables 1.1, 1.2, 1.3.

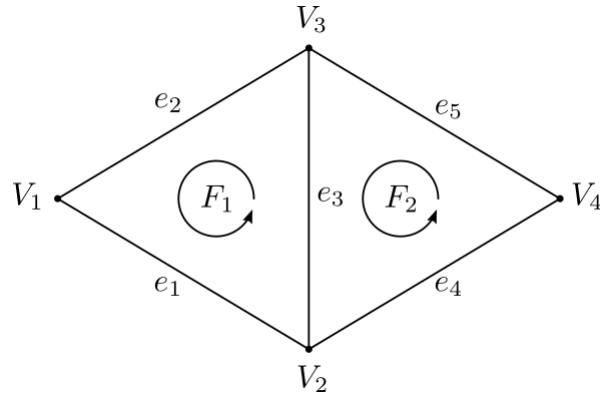


Figure 1.6: Example of winged edge representation.

| Edge | Vertex | | Face | | Left traverse | | Right traverse | |
|-------|---------|----------|-------|-------|---------------|-------|----------------|-------|
| | Initial | Terminal | Left | Right | Pred. | Succ. | Pred. | Succ. |
| e_1 | V_1 | V_2 | F_1 | | e_2 | e_3 | | |
| e_2 | V_3 | V_1 | F_1 | | e_3 | e_1 | | |
| e_3 | V_2 | V_3 | F_1 | F_2 | e_1 | e_2 | e_5 | e_4 |
| e_4 | V_2 | V_4 | F_2 | | e_3 | e_5 | | |
| e_5 | V_4 | V_3 | F_2 | | e_4 | e_3 | | |

Table 1.1: Edge table of a winged edge data structure for the figure 1.6.

| Vertex | Coordinates | | | Edge |
|--------|-------------|-------|-------|-------|
| | x | y | z | |
| V_1 | x_1 | y_1 | z_1 | e_1 |
| V_2 | x_2 | y_2 | z_2 | e_4 |
| V_3 | x_3 | y_3 | z_3 | e_2 |
| V_4 | x_4 | y_4 | z_4 | e_5 |

Table 1.2: Vertex table of a winged edge data structure for the figure 1.6.

| Face | Edge |
|-------|-------|
| F_1 | e_1 |
| F_2 | e_4 |

Table 1.3: Face table of a winged edge data structure for the figure 1.6.

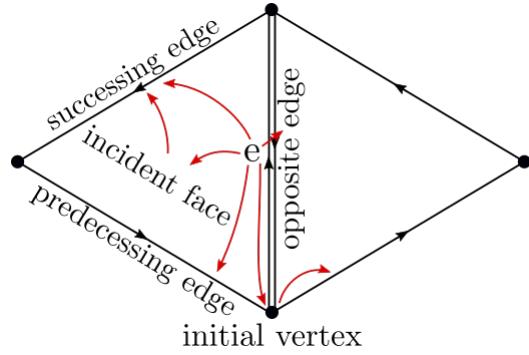


Figure 1.7: Visualisation of the half-edge data structure.

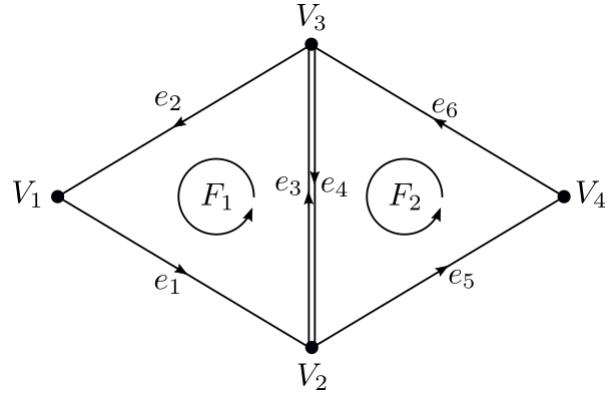


Figure 1.8: Example of half-edge representation.

In the half-edge representation, edges are split into two halves. Each half-edge stores reference to its initial vertex, left face, opposite half-edge and left traverse - predeccessing and successing half-edge. A visualisation of the half-edge is displayed on the figure 1.7. An example of half-edge representation is shown on the figure 1.8 and tables 1.4, 1.5, 1.6.

Range tree

Mesh structure

Triangular mesh consists of vertices, edges and faces. We use the half-edge data structure for mesh representation and the range tree for time efficient search of vertices in three dimensional interval. The mesh structure used for maintaining and modifying the triangular mesh consists of:

- list of vertices,
- list of half-edges,
- list of faces,
- set of active edges,
- set of checked edges,

| Edge | Vertex | Face | Edges | | |
|-------|--------|-------|---------|----------|----------------------------|
| | | | Initial | Incident | Pred. Succ. Opposite |
| e_1 | V_1 | F_1 | | e_2 | e_3 |
| e_2 | V_3 | F_1 | | e_3 | e_1 |
| e_3 | V_2 | F_1 | | e_1 | e_2 |
| e_4 | V_3 | F_2 | | e_6 | e_5 |
| e_5 | V_2 | F_2 | | e_4 | e_6 |
| e_6 | V_4 | F_2 | | e_5 | e_4 |

Table 1.4: Edge table of a half-edge data structure for the figure 1.8.

| Vertex | Coordinates | | | Edge |
|--------|-------------|-------|-------|---------------|
| | x | y | z | Outgoing edge |
| V_1 | x_1 | y_1 | z_1 | e_1 |
| V_2 | x_2 | y_2 | z_2 | e_3 |
| V_3 | x_3 | y_3 | z_3 | e_2 |
| V_4 | x_4 | y_4 | z_4 | e_6 |

Table 1.5: Vertex table of a half-edge data structure for the figure 1.8.

| Face | Edge |
|-------|-------|
| F_1 | e_1 |
| F_2 | e_4 |

Table 1.6: Face table of a half-edge data structure for the figure 1.8.

- set of bounding edges,
- range tree of all vertices.

Vertex consists of

- three coordinates,
- index of itself in the list of vertices,
- list of indices to all outgoing edges.

Half-edge consists of

- six coordinates,
- index of itself in the list of half-edges,
- index of initial vertex in the list of vertices,
- index of terminal vertex in the list of vertices,
- index of opposite edge in the list of half-edges,
- index of predecessing edge in the list of half-edges,
- index of successsing edge in the list of half-edges,
- index of incident face in the list of faces.

Face consists of

- nine coordinates,
- index of incident half-edge in the list of half-edges.

Implementation

1.6 CSG modelling for implicit surfaces

In this section we will discuss how constructive solid geometry can be used for modelling complex implicit surfaces. First major publication regarding constructive solid geometry was published in 1977 by Requicha and Voelcker [15]. More detailed mathematical foundations were published a year later, in 1978 by Requicha and Tilove [14].

Constructive solid geometry (CSG)

Constructive solid geometry is a technique used for modelling complex geometric objects using boolean operations on sets of points – union, intersection and difference. It gained popularity in 1980s as a powerful tool to create complex shapes from sets of geometrical primitives, such as cylinders, spheres and cones. The resulting model can be represented as a CSG tree, which contains geometrical primitives in its leaves and boolean operations in its internal nodes [5]. An example of such CSG tree can be seen on figure 1.9.

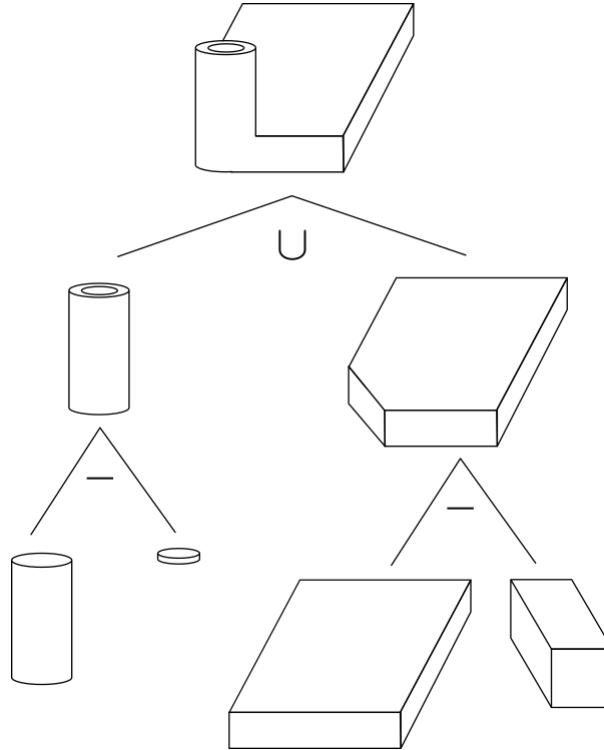


Figure 1.9: Example of CSG tree [5].

CSG for implicit surfaces

As the implicit surface divides space into inside ($F(x, y, z) < 0$) and outside ($F(x, y, z) > 0$), one can easily combine the idea of implicit surfaces and CSG modelling. Given two implicit surfaces represented by implicit functions F and G , the surface of the intersection of the interiors can be defined by implicit function $H = \min(F, G)$. Similarly, the surface of the union of the interiors can be defined by implicit function $H = \max(F, G)$ and lastly, the surface of the difference of the interiors can be defined by implicit function $H = \max(F, -G)$. Two dimensional example of this idea can be seen on figure 1.10.

Theorem 1 *The minimum function $\min(F, G)$ of two functions $F : \mathbb{R}^3 \rightarrow \mathbb{R}$ and $G : \mathbb{R}^3 \rightarrow \mathbb{R}$ is defined as*

$$\min(F, G) = F + G - \sqrt{F^2 + G^2}$$

. *The maximum function $\max(F, G)$ of two functions $F : \mathbb{R}^3 \rightarrow \mathbb{R}$ and $G : \mathbb{R}^3 \rightarrow \mathbb{R}$ is defined as*

$$\max(F, G) = F + G + \sqrt{F^2 + G^2}$$

These formulas allow us to model implicit surfaces which are the result of performing finite number of operations union, intersection and difference on arbitrary implicit functions.

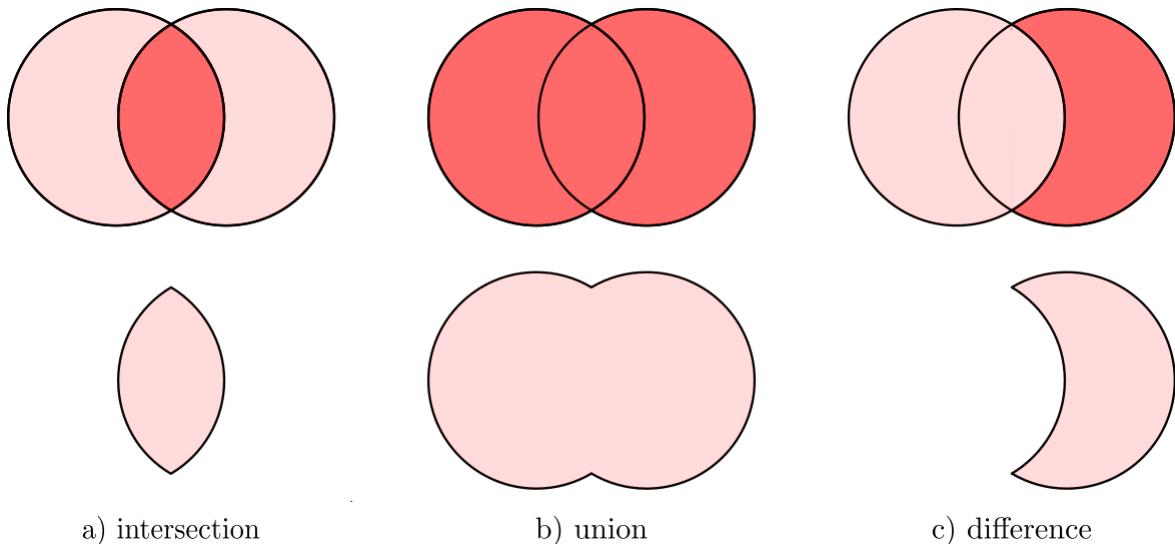


Figure 1.10: Boolean operations on implicit curves.

An easy example is creating an implicit equation representing a half of a sphere. Let us have an implicit function $F(x, y, z) = x^2 + y^2 + z^2 - 1$, which represents a unit sphere and another implicit function $G(x, y, z) = z$ which represents the xy plane. The minimum function of these two functions

$$\min(F, G) = x^2 + y^2 + z^2 - 1 + z - \sqrt{(x^2 + y^2 + z^2 - 1)^2 + z^2}$$

represents the surface of a half sphere. The visualisation of these surfaces can be seen on figure 1.11.

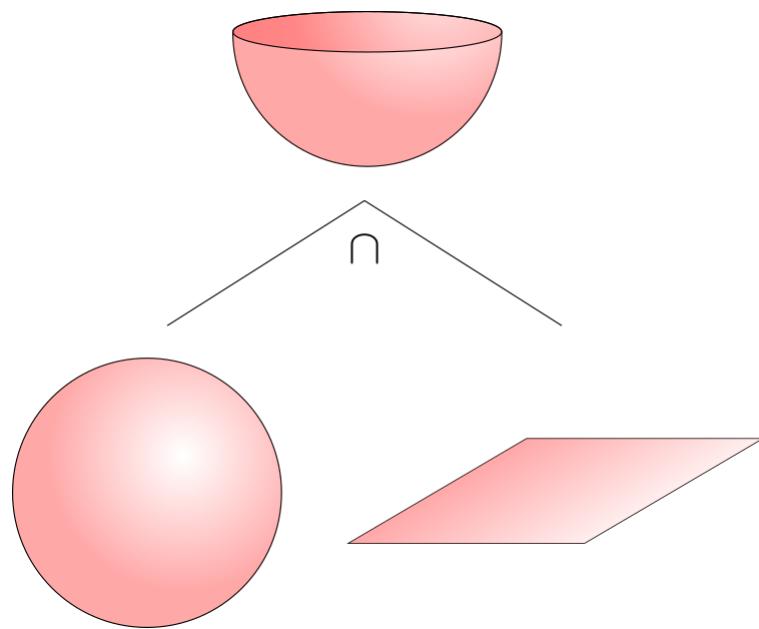


Figure 1.11: Intersection of a sphere and a plane.

Chapter 2

Our contributions

2.1 Triangulation adaptive to the local curvature

As we explained in the begining of chapter 1.1, curvature of the surface is a measure of how much the surface bends.

The triangulation of the surface should be accurate enough, but also memory efficient. This can be achieved by creating a triangulation which is locally adaptive to the curvature of the surface. Therefore having smaller triangles in the places where the surface is curved and having bigger triangles where surface is flatter.

In this section we present our implementation of the triangulation adaptive to the local curvature.

In the original algorithm, the height of the triangle which is projected to the surface is set to the constant value $\frac{\sqrt{3}}{2}e$, where e is the required length of the side of the triangle. To achieve the adaptivity of the triangles size, we set the height of the triangle to depend on the curvature in the given point, as shown in the image 2.1.

To identify the curved areas, we decided to use the maximal curvature. As the minimal and maximal curvature are both signed, we want to identify the areas depending on the absolute value of these curvatures.

We do not allow arbitrary height of the triangle to avoid edge-cases. We decided to restrict the allowed height to $\frac{1}{4}\frac{\sqrt{3}}{2}e > h > 4\frac{\sqrt{3}}{2}e$. We create a variable m —multiplicator, depending on κ_T and set the height of the new triangle to $h = m\frac{\sqrt{3}}{2}e$.

As κ_T has values in range $\langle 0, \infty \rangle$.

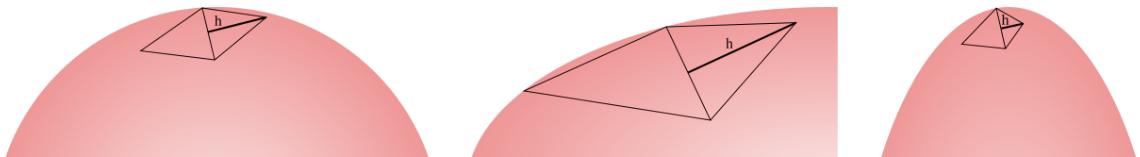


Figure 2.1: Adaptive height of the new triangle.

Definition 12 Let us define triangulation curvature of the surface S in the point $P = S(u, v)$ as $\kappa_T(u, v) = \max(|\kappa_{\min}|, |\kappa_{\max}|)$.

NOT FINISHED YET

2.2 Triangulation of ADE singularities

Analysis of the geometry of ADE singularities

ADE singularities are simple, isolated surface singularities, which can be expressed by corresponding implicit equations.

We already know, that A_{1--} singularity is locally represented as a cone. In this section we discuss geometric structure of other ADE surface singularities.

Definition 13 (TODO rewrite) Let us define branch of ADE singularity as the part of the surface, which is connected to the rest only by the singular point.

For our needs, we pick one triangulation vector for each branch of each ADE singularity. This triangulation vector is normalized vector either in the direction of rotation symmetry axis or an intersection of reflection symmetry planes of the corresponding branch. If the branch has only one reflection symmetry plane, the triangulation vector is picked to lie in the reflection symmetry plane.

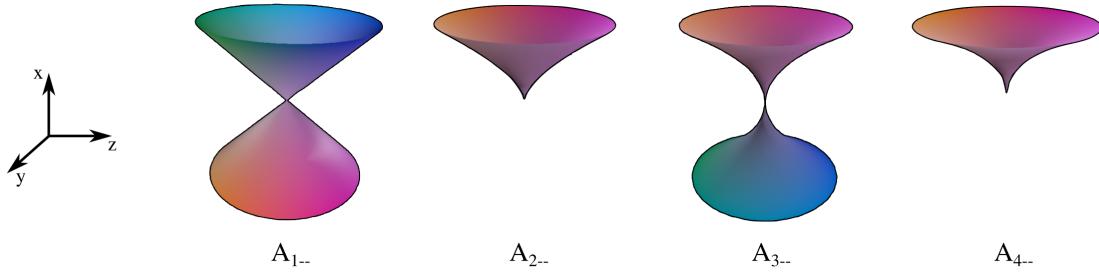
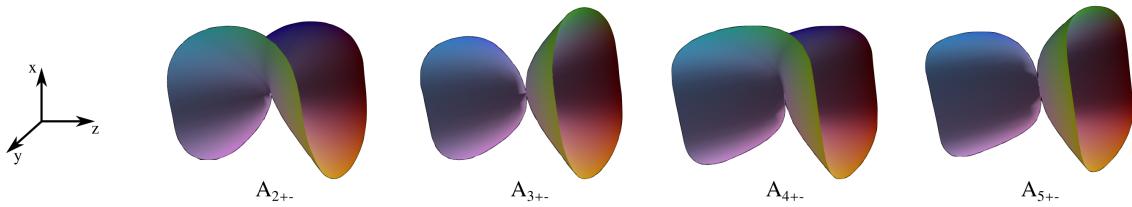
In the general case, triangulation vectors serve us as a partial information about the orientation of a singularity with respect to its normal form.

A_n singularities

As we can see from the equations $F(x, y, z) = x^{n+1} \pm y^2 \pm z^2$, A_{n+-} singularities are just rotated A_{n+-} singularities and A_{n++} singularities are a single point if n is odd and reflected A_{n--} singularities if n is even. We therefore only discuss geometry of A_{n--} and A_{n+-} singularities.

A_{n--} singularities are topologically equivalent to a cone if n is odd, therefore they have two branches. If n is even, they are topologically equivalent to a half cone or a plane, therefore they have a single branch. As n gets bigger, the tip of the cone gets sharper. As A_{n--} singularities are rotationally symmetrical, we pick the direction of axis of symmetry as triangulation vector. For a normal form, the triangulation vectors are $(1, 0, 0)$ (and $(-1, 0, 0)$ if n is odd). First four A_{n--} singularities can be seen on image 2.2.

A_{n+-} singularities are topologically equivalent to a cone if n is odd, therefore they have two branches. In the contrary with the previous singularities, as n gets bigger, the tip of the cone gets less sharp and flatter. Branches of these singularities have

Figure 2.2: A_{n--} singularities. [12]Figure 2.3: A_{n+-} singularities. [12]

reflection symmetry planes $x = 0$ and $y = 0$, therefore we pick the vectors $(0, 0, 1)$ and $(0, 0, -1)$ as the triangulation vectors.

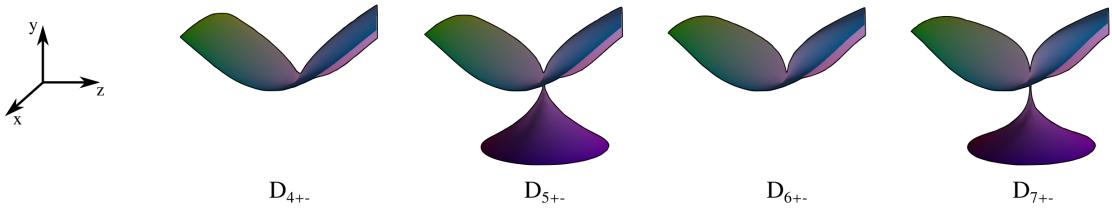
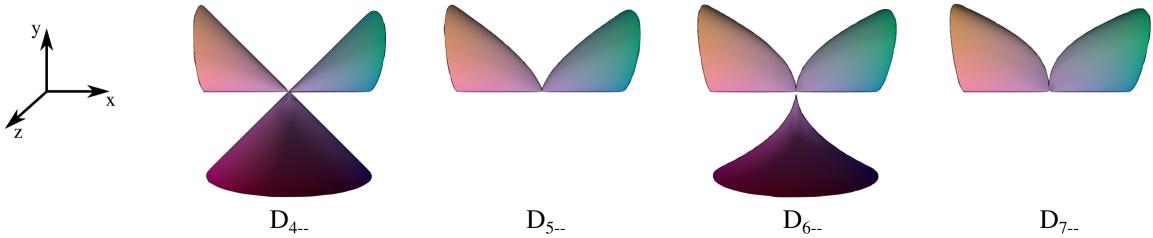
If n is even, A_{n+-} singularities are topologically equivalent to a plane with shape similar to hyperbolic paraboloid, therefore they have a single branch. First four A_{n+-} singularities can be seen on image 2.3. For this case, we pick the vector $(1, 0, 0)$ as a triangulation vector as these singularities have reflection symmetry planes $y = 0$ and $z = 0$.

D_n singularities

Given by equations $F(x, y, z) = yx^2 \pm y^{n-1} \pm z^2$, we consider 8 categories. For given sign combination and parity of n , the singularities are topologically equivalent, with sharper(or flatter) features around the singularities for increasing value of n similar to A_n singularities.

We can therefore say that D_n singularities can be classified into 8 categories locally represented by the following equations:

- D_{4++} $yx^2 + y^3 + z^2$
- D_{5++} $yx^2 + y^4 + z^2$
- D_{4+-} $yx^2 + y^3 - z^2$
- D_{5+-} $yx^2 + y^4 - z^2$
- D_{4-+} $yx^2 - y^3 + z^2$

Figure 2.4: D_{n+-} singularities. [12]Figure 2.5: D_{n--} singularities. [12]

- D_{5-+} $yx^2 - y^4 + z^2$
- D_{4--} $yx^2 - y^3 - z^2$
- D_{5--} $yx^2 - y^4 - z^2$.

Now we look at some equivalences between these 8 categories. D_{4++} singularity is reflected D_{4+-} singularity. D_{5++} singularity is reflected D_{5--} singularity. D_{5-+} singularity is reflected D_{5+-} singularity. D_{4-+} singularity is reflected D_{4--} singularity.

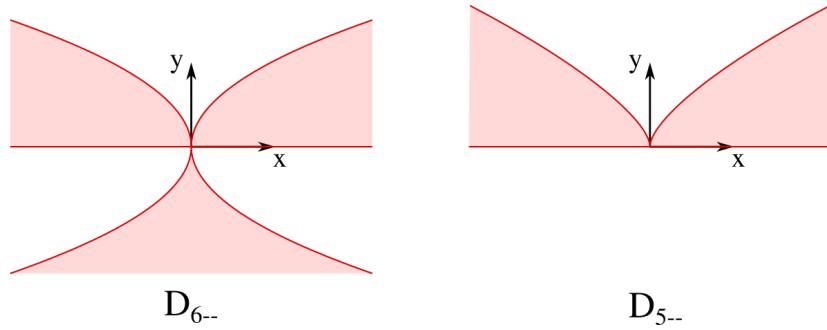
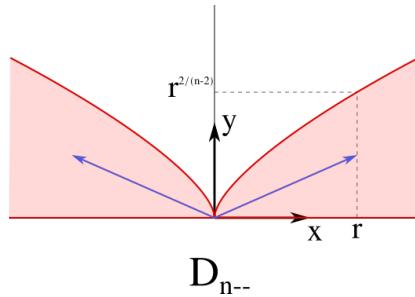
We therefore only analyze geometry of D_{n+-} singularities and D_{n--} singularities.

D_{n+-} singularities are topologically equivalent to a plane when n is even and to a cone when n is odd. Again, as n gets bigger, the features around singularities get sharper. Symmetry planes of these singularities are $x = 0$ and $z = 0$, therefore we pick $(0, 1, 0)$ (and $(0, -1, 0)$ when n is odd) as trinauglation vectors. First four D_{n+-} singularities can be seen on image 2.4.

D_{n--} singularities are topologically equivalent to a cone when n is odd and to a 3 halfcones connected in the singular point when n is even. First four D_{n--} singularities can be seen on image 2.5.

Symmetry plane for all branches of these singularities is $z = 0$. the intersection of the surface and plane $z = 0$ is displayed on image 2.6.

For D_{n--} singularity, the intersections of the two branches where $y \geq 0$ are bounded by curves $y = 0$ and $x^2 = y^{n-2}$. For given r , we pick the triangulation vectors as $(r, \frac{1}{2}r^{\frac{2}{n-2}}, 0)$ and $(-r, \frac{1}{2}r^{\frac{2}{n-2}}, 0)$. The resulting vectors are displayed on image 2.7 by blue arrow. Parameter r is changed based on the length of the edge of triangulation

Figure 2.6: Intersection of D_{n--} singularities with plane $z = 0$.Figure 2.7: Triangulation vectors for two branches of D_{n--} singularities.

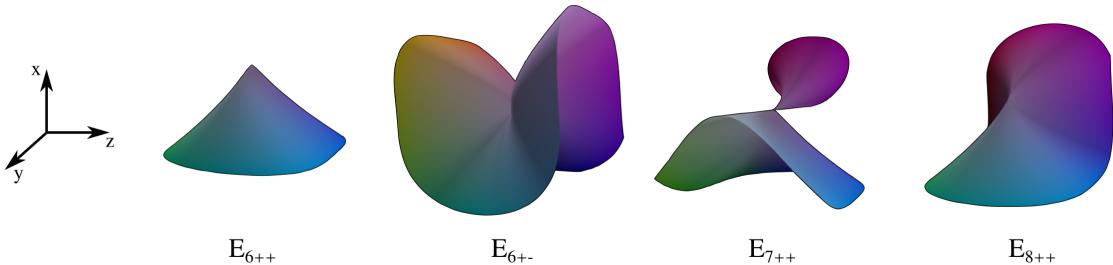
triangle. displayed on image 2.7 by blue arrow. Parameter r is changed based on the length of the edge of triangulation triangle.

The third branch where $y \leq 0$ has another plane of symmetry $x = 0$, therefore triangulation vector for this branch is chosen as $(0, -1, 0)$.

E_6, E_7 and E_8 singularities

Given by equations $F(x, y, z) = x^3 \pm y^4 \pm z^2$, $F(x, y, z) = x^3 \pm xy^3 \pm z^2$ and $F(x, y, z) = x^3 \pm y^5 \pm z^2$, we can see the following equivalences: E_{6++} singularity is reflected E_{6--} singularity. E_{6+-} singularity is reflected E_{6-+} singularity. E_{7+-}, E_{7-+} and E_{7--} are all reflected E_{7++} singularity. E_{8+-}, E_{8-+} and E_{8--} are all reflected E_{8++} singularity.

We only analyze geometry of E_{6++} , E_{6+-} , E_{7++} and E_{8++} singularities. These singularities are displayed on the image 2.8.

Figure 2.8: E_n singularities. [12]

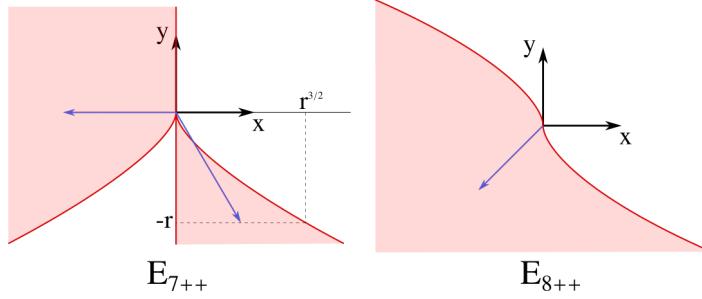


Figure 2.9: Intersection of E_{7++} and E_{8++} singularities with plane $z = 0$.

Both E_{6++} and E_{6+-} are topologically equivalent to a plane, thus they each have only one branch. The planes of symmetry of both of these branches are $y = 0$ and $z = 0$, therefore we pick $(-1, 0, 0)$ as the triangulation vector.

E_{7++} singularity is topologically equivalent to a cone, therefore it has two branches. The plane of symmetry of this singularity is $z = 0$.

E_{8++} singularity is also topologically equivalent to a plane, therefore it has only one branch. This branch has only one plane of symmetry $z = 0$.

We again look at the intersection of the surfaces with the plane of symmetry, this is displayed on image 2.9.

For E_{7++} singularity, we pick $(-1, 0, 0)$ and $(\frac{1}{2}r^{\frac{3}{2}}, -r, 0)$ as triangulation vectors. For E_{8++} singularity, we pick $(-1, -1, 0)$ as a triangulation vector. These vectors are displayed on the image 2.9 as blue arrows.

Analytical calculation of local triangulation of some ADE singularities

For given edge size e , we want to calculate the local triangulation of ADE singularities, such that edges on the border of the local triangulation have length e .

A_{n--} singularities

For A_{n--} singularities, we create a disc of 6 isosceles triangles with vertex in the singular point. The bases of these triangles create regular hexagon in the plane P parallel to the plane $x = 0$, as showed on the image 2.10. Given by equation $x^{n+1} - y^2 - z^2 = 0$, we find the distance of the plane P from the plane $x = 0$ for the given length e of the sides of the hexagon.

Let e be the length of the side of the hexagon, then the circumscribed circle has radius e . This circle is identical with the intersection of the surface and the plane $x = h$. The equation of the intersecting circle is $y^2 + z^2 = h^{n+1}$ therefore, the radius can be also expressed as $r = h^{\frac{n+1}{2}}$, which emerges $h = e^{\frac{2}{n+1}}$. Knowing the distance of the plane, one can easily calculate the length of the arms of the triangles using

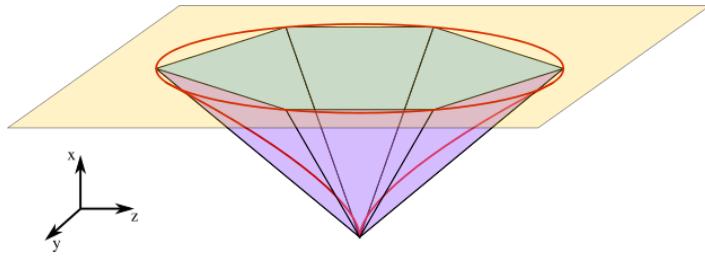
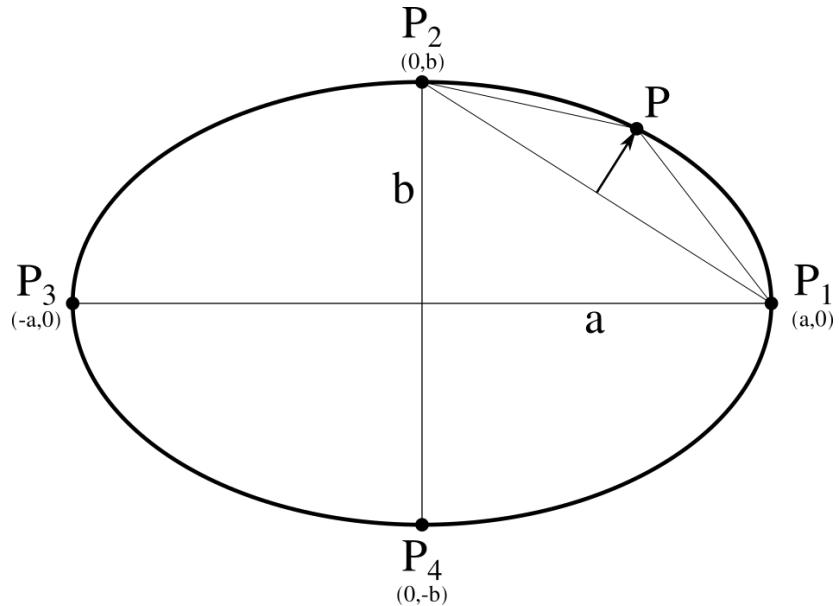
Figure 2.10: Triangulation of A_{n--} singularity.

Figure 2.11: Equidistant points on ellipse.

Pythagorean theorem:

$$a^2 = h^2 + e^2 \implies a = \sqrt{e^{\frac{4}{n+1}} + e^2}$$

D_n singularities

Some D_n singularities have branches with elliptical intersection with a plane parallel to the plane $y = 0$. As ellipses have 2 axes of symmetry, we create 8 triangles for these branches.

Let us have an ellipse E with semi-major axis a and semi-minor axis b . We are going to create eight triangles with apex in the singular point. The other points of the triangles lie on the ellipse and they have the same length of the base.

As displayed on image 2.11, we pick the leftmost, the rightmost, the top and the bottom points. Then we can calculate the point P on ellipse equidistant from points P_1 and P_2 .

$$\frac{1}{2}(a, b) + \frac{t}{2}(b, a) \in E \implies \frac{(a + tb)^2}{4a^2} + \frac{(b + ta)^2}{4b^2} = 1$$

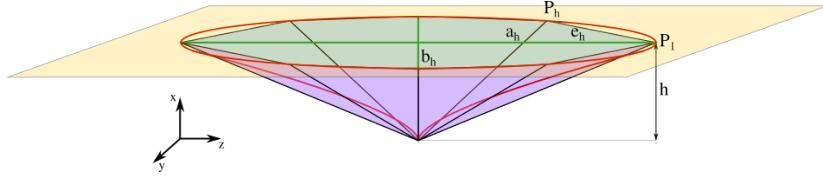


Figure 2.12: TODO.

$$4b^2(a^2 + 2atb + t^2b^2) + 4a^2(b^2 + 2atb + t^2a^2) - a^2b^2 = 0$$

$$4(b^4 + a^4)t^2 + 8ab(b^2 + a^2)t + 7a^2b^2 = 0$$

$$t = \frac{ab(\sqrt{3a^4 + 2a^2b^2 + 3b^4} - a^2 - b^2)}{a^4 + b^4}$$

$$P = \frac{1}{2}(a, b) + \frac{ab(\sqrt{3a^4 + 2a^2b^2 + 3b^4} - a^2 - b^2)}{2(a^4 + b^4)}(b, a)$$

TODO binary search for height

Given edge length e , we are not able to calculate the height in which the distance between points P_1 and P is e . The visualization showing this is on the image 2.12.

We use binary search to find such height. Given the height and the singularity class, we can calculate the semi-major axis and semi-minor axis as

$$D_{n+-} : -hx^2 + h^{n-1} - z^2 = 0 \quad h > 0$$

$$x^2 + \frac{z^2}{h} = h^{n-2}$$

$$\frac{x^2}{h^{n-2}} + \frac{z^2}{h^{n-1}} = 1 \implies a_h = \max(h^{\frac{n-2}{2}}, h^{\frac{n-1}{2}}) \wedge b_h = \min(h^{\frac{n-2}{2}}, h^{\frac{n-1}{2}}).$$

As we can see, we get the same ellipse for D_{n--} singularities:

$$D_{n--} : -hx^2 - h^{n-1} - z^2 = 0 \quad h > 0$$

$$2|n \wedge x^2 + \frac{z^2}{h}| = -h^{n-2} \implies x^2 + \frac{z^2}{h} = h^{n-2}.$$

Then

$$P_h = \frac{1}{2}(h^{\frac{n-2}{2}}, h^{\frac{n-1}{2}}) + \frac{h^{\frac{2n-3}{2}}(\sqrt{3h^{2n-4} + 2h^{2n-3} + 3h^{2n-2}} - h^{n-2} - h^{n-1})}{2(h^{2n-4} + h^{2n-2})}(h^{\frac{n-1}{2}}, h^{\frac{n-2}{2}})$$

$$P_h = \frac{1}{2}(h^{\frac{n-2}{2}}, h^{\frac{n-1}{2}}) + \frac{h^{\frac{1}{2}}(\sqrt{3 + 2h + 3h^2} - 1 - h)}{2(1 + h^2)}(h^{\frac{n-1}{2}}, h^{\frac{n-2}{2}})$$

and we can calculate $e_h = \|P_h - P_1\|$.

As $e \leq a_h$, we can start the binary search on the interval $\langle 0, a^{\frac{2}{n-2}} \rangle$ or $\langle 0, a^{\frac{2}{n-1}} \rangle$ and finish, when required precision is reached.

TODO want to try to prove that $\|P_h - P_1\|$ is monotone in h .

Numerical calculation of local triangulation of ADE singularities

For other types of ADE singularities, the exact analytical calculations become more complicated. In this section we present an approach for triangulation of all types of ADE singularities using numerical algorithms such as binary search.

We begin by dividing ADE singularities into three categories.

1. Singularities topologically equivalent to a plane:

- A_{n--} , where $2|n$,
- A_{n+-} , where $2|n$,
- D_{n+-} , where $2|n$,
- E_{6++} ,
- E_{6+-} ,
- E_{8++} .

2. Singularities topologically equivalent to a cone except E_{7++} :

- A_{n--} , where $2 \nmid n$,
- A_{n+-} , where $2 \nmid n$,
- D_{n+-} , where $2 \nmid n$,
- D_{n--} , where $2 \nmid n$

3. Other singularities:

- D_{n--} , where $2|n$,
- E_{7++} .

We propose a general solution for first and second categories and we treat the singularities in the third category as a special cases.

Singularities topologically equivalent to a plane

Singularities topologically equivalent to a cone except E_{7++}

Other singularities - special cases

Triangulation of a plane with multiple A_{n--} singularities

In this section, we present an approach for creating an implicit equation of a surface which consists of a plane and arbitrary many A_{n--} singularities C^1 smoothly connected to this plane.

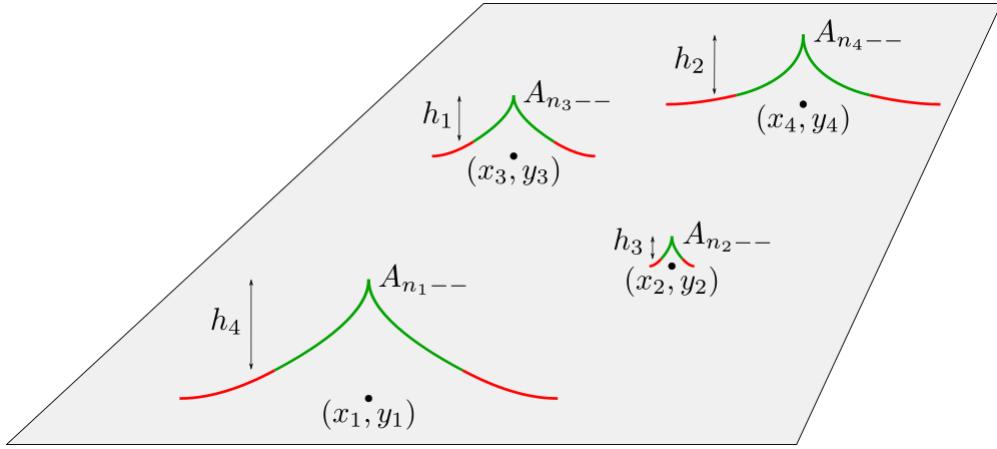


Figure 2.13: Plane with singularities.

Input and output

In this section, the following data are provided on the input:

1. the number of singularities - m ,
2. m discrete points on a plane - $(x_1, y_1), \dots, (x_m, y_m)$,
3. m degrees of the singularities - n_1, \dots, n_m ,
4. m heights at which each singularity is connected - h_1, \dots, h_m .

The visualisation of desired output function can be seen on the figure 2.13. On this figure, the singularity is displayed by green color, the red color is used to display the function which connects the singularity to a plane - the bump function. There are some limitations on the input data. As we do not want the singularities or the bump functions to intersect, we require that each pair of input points is distanced d_{ij} from each other. We specify the value of d_{ij} in the section TODO.

Bump function

Definition 14 *The support $\text{supp}(f)$ of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a set of points where f is not zero:*

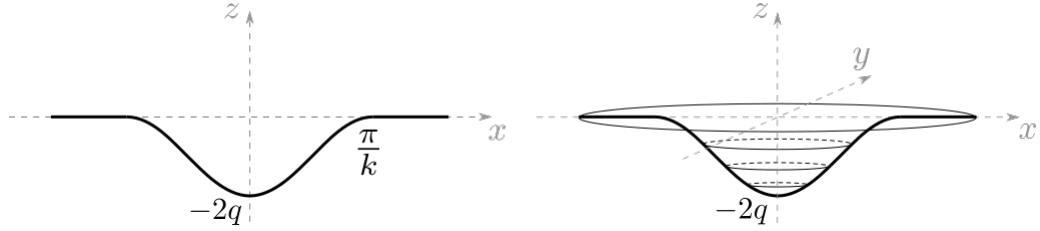
$$\text{supp}(f) = \{x \in \mathbb{R}^n : f(x) \neq 0\}.$$

The closed support of the function f is defined as a closure of $\text{supp}(f)$.

Bump function if a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ which is smooth (C^∞) and compactly supported (the closed support of the function f is a compact subset of \mathbb{R}^n).

The most common example of such bump function is the function

$$f(x) = \begin{cases} e^{-\frac{1}{1-x^2}}, & x \in (-1, 1) \\ 0, & \text{otherwise,} \end{cases}$$

Figure 2.14: C^1 cosine bump function.

| Funcion name | Implicit equation |
|-------------------------|--|
| Rotated cosine function | $x + q \cdot \cos(k \cdot \sqrt{(y - p_y)^2 + (z - p_z)^2}) + q = 0$ |
| Cylinder | $(y - p_y)^2 + (z - p_z)^2 - (\frac{\pi}{k})^2 = 0$ |
| Plane | $x=0$ |

Table 2.1: Implicit equations for bump function modeling.

which is both C^∞ and compactly supported.

The bump function can be used to smoothly connect a curve to a line or a surface to a plane in higher dimension. If we only need to connect a curve and a line C^n smoothly, we only need a bump function which connects C^n smoothly to a line.

In our work, we connect two surfaces with C^1 continuity and for this purpose we use the function

$$f(x) = \begin{cases} -q \cdot \cos(k \cdot x) - q, & x \in (-\frac{\pi}{k}, \frac{\pi}{k}) \\ 0, & \text{otherwise,} \end{cases}$$

rotated about z -axis. The result of the rotation is the following function:

$$f(x, y) = \begin{cases} -q \cdot \cos(k \cdot (x^2 + y^2)) - q, & x^2 + y^2 \leq (\frac{\pi}{k})^2 \\ 0, & \text{otherwise.} \end{cases}$$

Both of these functions can be seen on the figure 2.14.

Implicit equation of the cosine bump function

To construct the implicit equation of the cosine bump function, we use CSG - constructive solid geometry, which is described in the section 1.6.

First, we cut out the part of the rotated cosine, where $x^2 + y^2 < (\frac{\pi}{k})^2$ using cylinder and intersection operation. Next, we use a plane and the union operation to *glue* the bump to the plane. The described process is displayed on the figure 2.15 in the form of CSG tree. We use the following equations of the surfaces to model the cosine bump function:

Parameters q and k allow us to change the amplitude and the frequency of the cosine function, parameters p_y and p_z are used to move the bump function to the given point (p_y, p_z) .

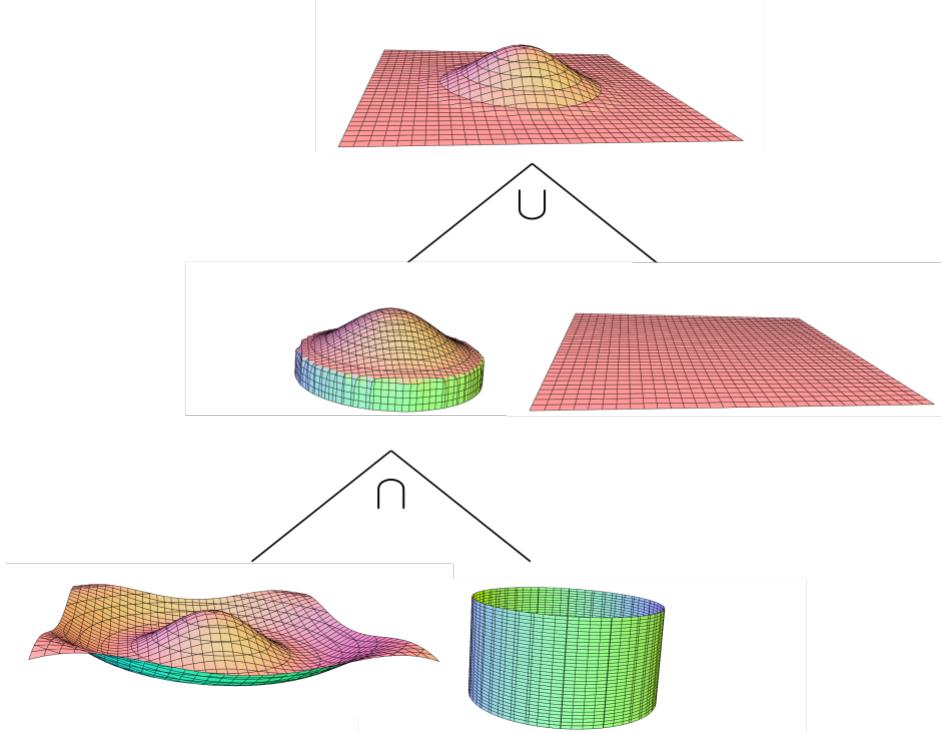


Figure 2.15: Construction of the cosine bump function using CSG.

Attaching singularities to the plane using the cosine bump function

Given the type of the singularity - n and given height - h , we calculate the constants of the cosine bump function to connect C^1 smoothly to the given singularity.

The singularity given by the implicit equation $x^{n+1} - y^2 - z^2$ intersected with the plane $x = h$ produces a circle with the radius $r = \sqrt{h^{n+1}}$. The cosine bump function is scaled using q and k to smoothly connect the singularity in the middle of the cosine bump function. This approach is displayed on the figure 2.16.

As the singularity is attached in the middle of the bump function, we get the equality $r = \frac{\pi}{2k}$ and therefore $\sqrt{h^{n+1}} = \frac{\pi}{2k} \implies k = \pi/(2\sqrt{h^{n+1}})$. The parameter q is calculated from C^1 continuity requirement. We require the gradients to be linearly dependent in the points of connection. Due to the rotation symmetry, we check it only for the intersection with the plane $z = 0$ for the point $(-q, \frac{\pi}{2k})$.

$$F = (x + h + q)^{n+1} - y^2 \implies \nabla F = [(n+1)(x + h + q)^n, -2y]$$

$$\nabla F \left(-q, \frac{\pi}{2k} \right) = \left[(n+1)h^n, -\frac{\pi}{k} \right]$$

$$G = x + q \cdot \cos(ky) + q \implies \nabla G = [1, -qk \cdot \sin(ky)]$$

$$\nabla G \left(-q, \frac{\pi}{2k} \right) = [1, -qk]$$

Requiring $\nabla F(-q, \frac{\pi}{2k}) = s \cdot \nabla G(-q, \frac{\pi}{2k})$ and knowing $k = \pi/(2\sqrt{h^{n+1}})$, we get $s = (n+1)h^n$ and therefore $q = 4h/(\pi(n+1))$.

2.3. TRIANGULATION OF NON-ISOLATED SINGULARITIES OF TRANSLATION SURFACES

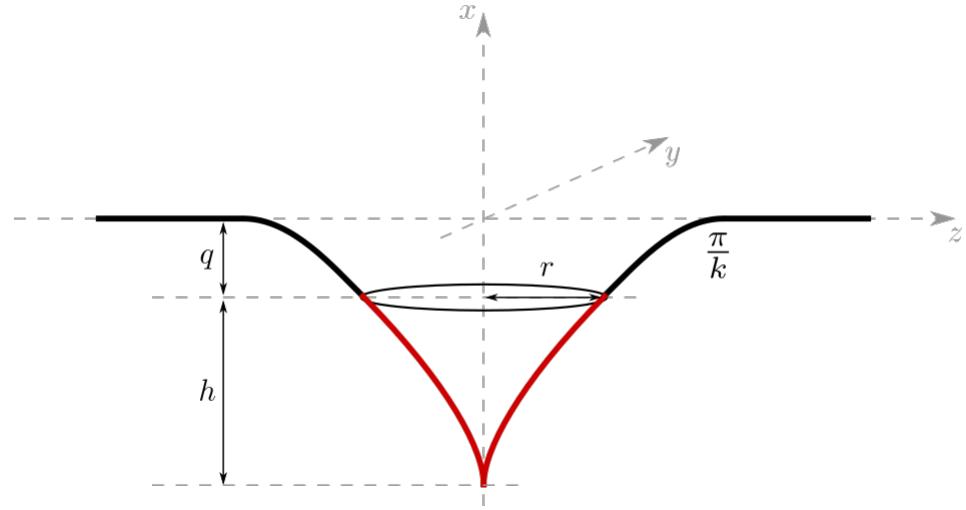


Figure 2.16: Attaching the singularity to a plane using the cosine bump function.

After calculating the parameters q and k of the cosine bump function, we proceed to connect the singularity to the bump function. We use intersection with the plane $x = -q$ to get the sections of the singularity and the section of the bump function and lastly, we use union of these two surfaces. The described process is displayed on the figure 2.17. Curious reader can find the detailed calculation of the implicit equation in the appendix 4.

To connect multiple singularities to the same plane, we construct the implicit equation for each of these singularities and then use the union operation to create a surface with multiple singularities. This procedure is displayed on the image 2.18.

Limitations on the input data

As we already mentioned, we require that each pair of input points is distanced d_{ij} from each other. As the radius of the closed support of the cosine bump function is $r = \frac{\pi}{k} = 2\sqrt{h^{n+1}}$, the distance between two input points p_j, p_j must be at least $d_{ij} = 2\sqrt{h_i^{n+1}} + 2\sqrt{h_j^{n+1}}$. This way, both singularities and the corresponding bump functions do not intersect.

Visualization of results of the triangulated surfaces

2.3 Triangulation of non-isolated singularities of translation surfaces

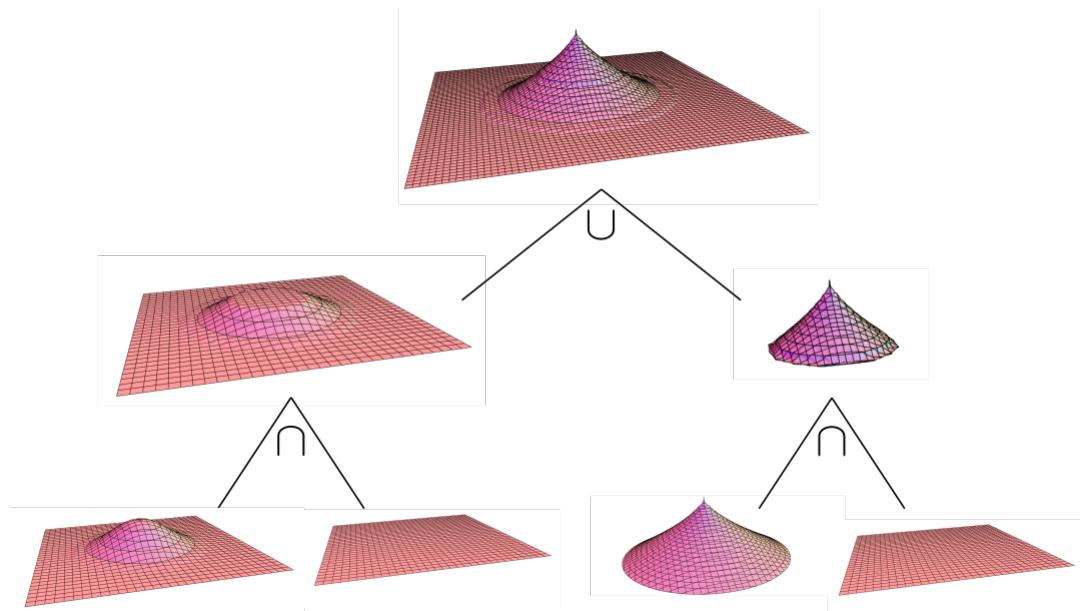


Figure 2.17: Attaching the singularity to a plane using CSG.

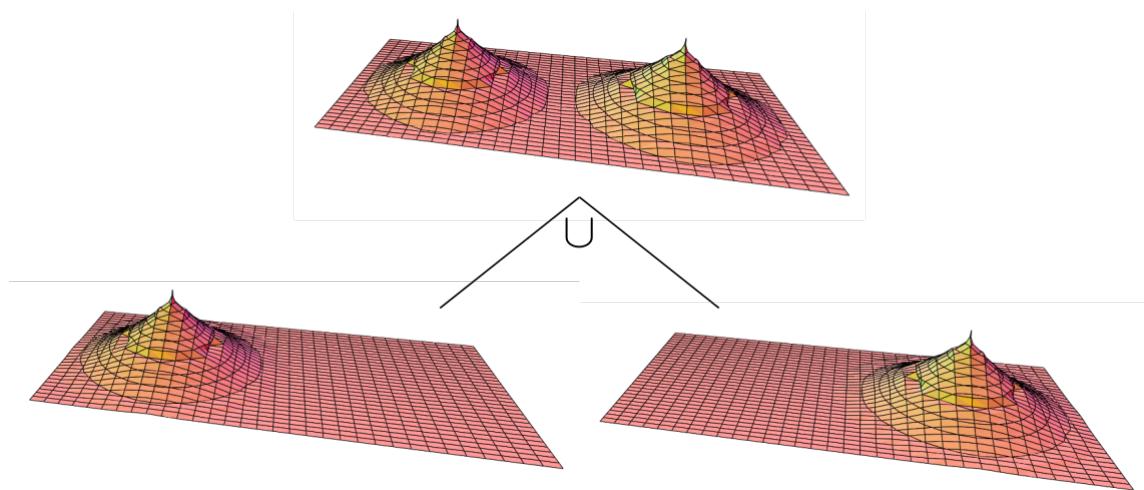


Figure 2.18: Plane with multiple attached singularities.

Chapter 3

Results

3.1 Quality criteria

3.2 Comparison with TODO

Chapter 4

Future work

Conclusion

TODO Conclusion

Bibliography

- [1] Normal forms for functions near degenerate critical points, the weyl groups of a k, d k, e k and lagrangian singularities. *Functional Analysis and its applications*, 6:254–272, 1972.
- [2] Samir Akkouche and Eric Galin. Adaptive implicit surface polygonization using marching triangles. In *Computer Graphics Forum*, volume 20, pages 67–80. Wiley Online Library, 2001.
- [3] Bruce G Baumgart. A polyhedron representation for computer vision. In *Proceedings of the May 19-22, 1975, national computer conference and exposition*, pages 589–596, 1975.
- [4] Evgenii Borisovich Dynkin. The structure of semi-simple algebras. *Uspekhi Matematicheskikh Nauk*, 2(4):59–127, 1947.
- [5] James D Foley, Foley Dan Van, Andries Van Dam, Steven K Feiner, and John F Hughes. *Computer graphics: principles and practice*, volume 12110. Addison-Wesley Professional, 1996.
- [6] Ron Goldman. Curvature formulas for implicit curves and surfaces. *Computer Aided Geometric Design*, 22(7):632–658, 2005.
- [7] M Hazewinkel, W Hesselink, D Siersma, and FD Veldkamp. The ubiquity of coxeter-dynkin diagrams. *Nieuw Arch. Wisk*, 25:257–307, 1977.
- [8] Adrian Hilton, Andrew J Stoddart, John Illingworth, and Terry Windeatt. Marching triangles: range image fusion for complex object modelling. In *Proceedings of 3rd IEEE international conference on image processing*, volume 2, pages 381–384. IEEE, 1996.
- [9] Lutz Kettner. Using generic programming for designing a data structure for polyhedral surfaces. *Computational Geometry*, 13(1):65–90, 1999.
- [10] PhD. Kristína Korecová, doc. RNDr. Pavel Chalmovianský. Algoritmy triangulácie implicitne definovanej plochy. 2021.

- [11] George S Lueker. A data structure for orthogonal range queries. In *19th Annual Symposium on Foundations of Computer Science (sfcs 1978)*, pages 28–34. IEEE, 1978.
- [12] Dr. Richard Morris. 3d viewer-generator.
- [13] TIAGO NOVELLO, VINÍCIUS DA SILVA, GUILHERME SCHARDONG, LUIZ SCHIRMER, HÉLIO LOPES, and LUIZ VELHO. Differential geometry of implicit surfaces. 2021.
- [14] Aristides Requicha and Robert Tilove. Mathematical foundations of constructive solid geometry: General topology of closed regular sets. 1978.
- [15] Aristides AG Requicha and Herbert B Voelcker. Constructive solid geometry. 1977.
- [16] Michael Spivak. *A comprehensive introduction to differential geometry*, volume 3. Publish or Perish, Incorporated, 1975.
- [17] Kevin Weiler. Edge-based data structures for solid modeling in curved-surface environments. *IEEE Computer graphics and applications*, 5(1):21–40, 1985.

Appendix A

$$f_1 = (x - p_x)^{n+1} - (y - p_y)^2 - (z - p_z)^2$$

$$f_2 = -x - q$$

$$f_3 = f_1 \cap f_2$$

$$f_3 = f_1 + f_2 - \sqrt{f_1^2 + f_2^2}$$

$$f_4 = x + q + q \cdot \cos \left(k \sqrt{(y - p_y)^2 + (z - p_z)^2} \right)$$

$$f_5 = 4h^{n+1} - (z - p_z)^2 - (y - p_y)^2$$

$$f_6 = f_4 \cap f_5$$

$$f_6 = f_4 + f_5 - \sqrt{f_4^2 + f_5^2}$$

$$f_7 = x$$

$$f_8 = f_6 \cup f_7$$

$$f_8 = f_6 + f_7 + \sqrt{f_6^2 + f_7^2}$$

$$f_9 = x + q$$

$$f_{10} = f_8 \cap f_9$$

$$f_{10} = f_8 + f_9 - \sqrt{f_8^2 + f_9^2}$$

$$f_{11} = f_{10} \cup f_3$$

$$f_{11} = f_{10} + f_3 + \sqrt{f_{10}^2 + f_3^2}$$

$$f_{11} =$$

The corresponding surfaces are displayed on the figure 4.1

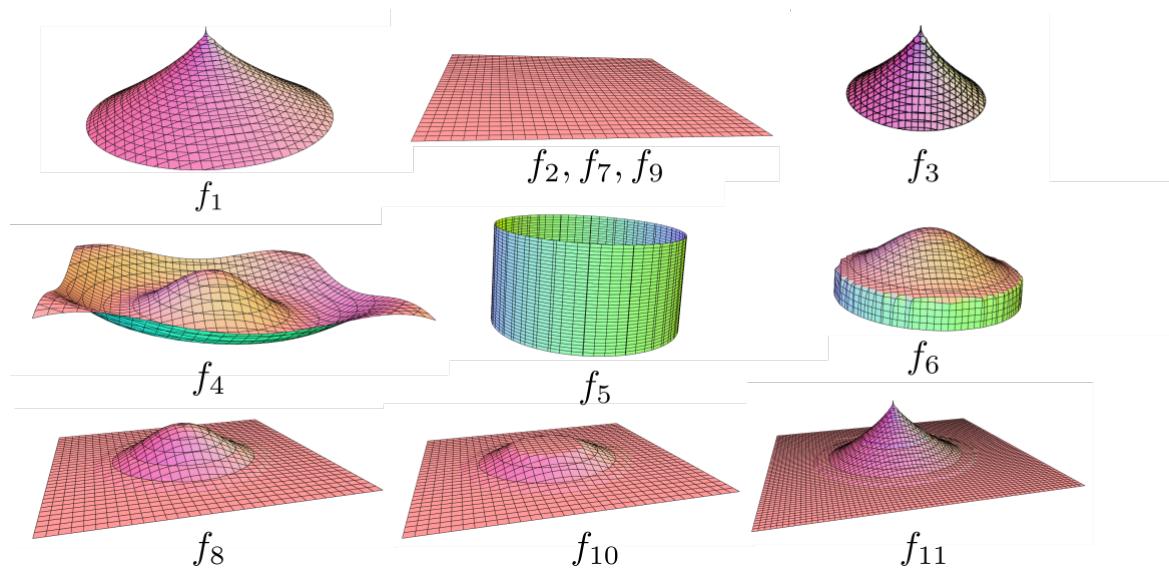


Figure 4.1: Attaching the singularity to a plane using CSG.

Appendix B