max planck institut
informatik

Eric Berberich

CGGC WS2009/2010: Applications

Max Planck Institute for Informatics

February 1, 2010

Two-dimensional arrangements

max planck institut informatik

Definition

For a set of curves $\mathcal{C}$ in $\mathcal{D}$ with $dim(\mathcal{D}) = 2$, the *arrangement* $\mathcal{A}(\mathcal{C})$ is the partitioning of $\mathcal{D}$ into cells of dimension 0, 1, and 2 induced by $\mathcal{C}$. Cells are called *vertices*, *edges*, and *faces*.

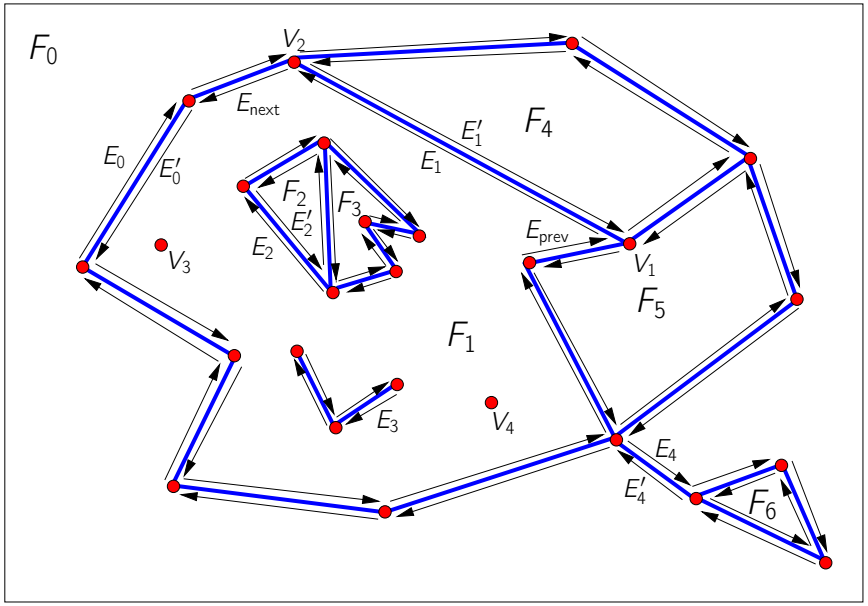
Fundamental structure in Computational Geometry:

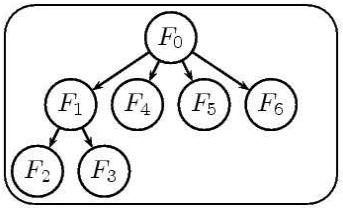
- computer vision
- geographic information systems
- robot motion planning
- we use it, e.g.,
  - for minimization diagram
  - as 2D-structure for stratification of surfaces
- model geometric problems in "arrangement"-lingo, e.g., using duality, or configuration space
- enables traversal & queries
- represents continuous problem in discretized (combinatorial) chunks

Representation as DCEL

max planck institut informatik

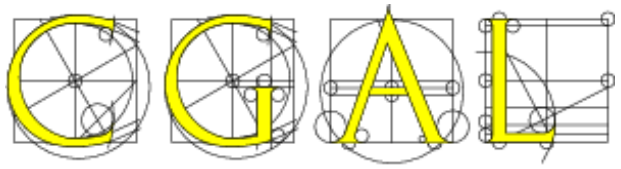
Doubly-connected-edge-list:





- Plane: Each face has ≤ 1 outer (counter-clockwise) boundary cycle and a ≥ 0 of inner (clockwise) boundary cycles
- Outer and inner cycles define *nesting graph*. Plane: tree

`Arrangement_2` package of

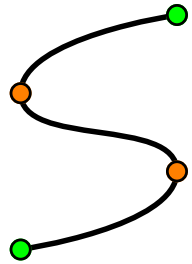


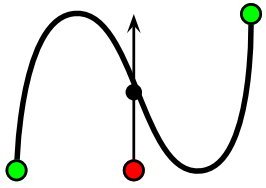
- constructs, maintains, modifies, traverses, queries arrangements of bounded curves in the plane (v3.2)
- modular due to generic programming
- efficient and robust, if used with exact geometric operations
- implements generic sweep line/zone algorithm, that
  - handle all degeneracies: e.g., vertical curves, multiple curves running through a common point, etc.
  - use *visitor* pattern to decouple *combinatorics* of sweep/zoning from *output*, e.g. reporting intersections or constructing DCEL
- `Arrangement_2<GeoTraits,...>`
  - parameter of the data structures and algorithms
  - defines the family of curves that induce the arrangement
  - must fulfill *ArrangementTraits_2* concept

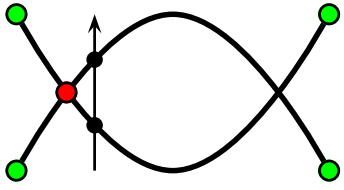
Geometric operations

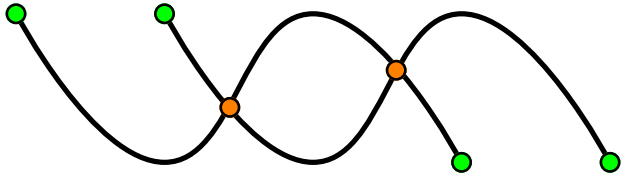
max planck institut informatik

- Types: `Curve_2`, `X_monotone_curve_2`, `Point_2`
- Methods:
  - Subdivide a curve into x-monotone curves
  - Compare two points lexicographically
  - Determine the relative position of a point and an x-monotone curve
  - Determine the relative position of two x-monotone curves to the right (or left) of a point
  - Find all intersections of two x-monotone curves







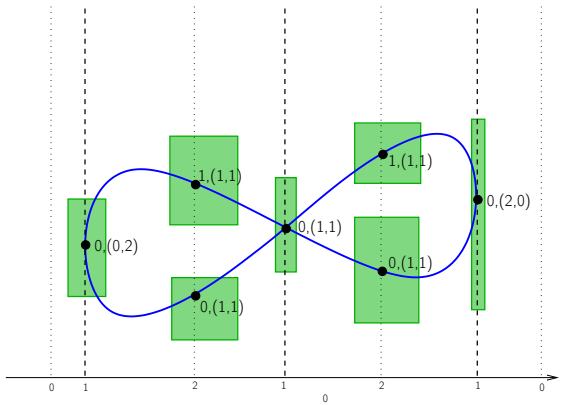


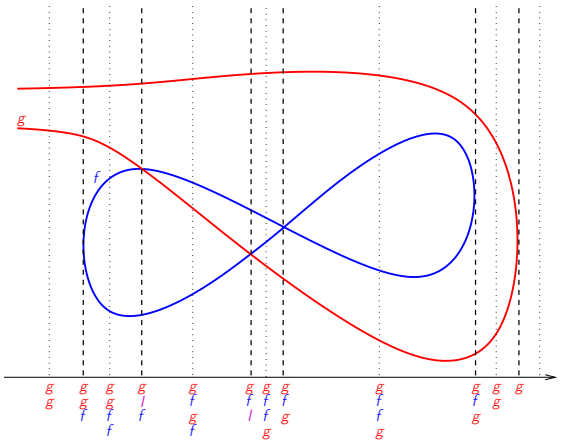
Arrangements of algebraic curves

max planck institut informatik

- was assumed to be impossible only a few years ago
- uses CGAL's `Algebraic_curve_kernel_2`
  efficient by extensive use of approximative, but certified methods

[Eigenwillig, Kerber, Wolpert 07/08]





- `Curved_kernel_via_analysis_2` rewrites analyses into geometric operations (GeometryTraits class) [B./Emeliyanenko 08]
- supports curves of arbitrary degree and all degeneracies

Arrangements of algebraic curves

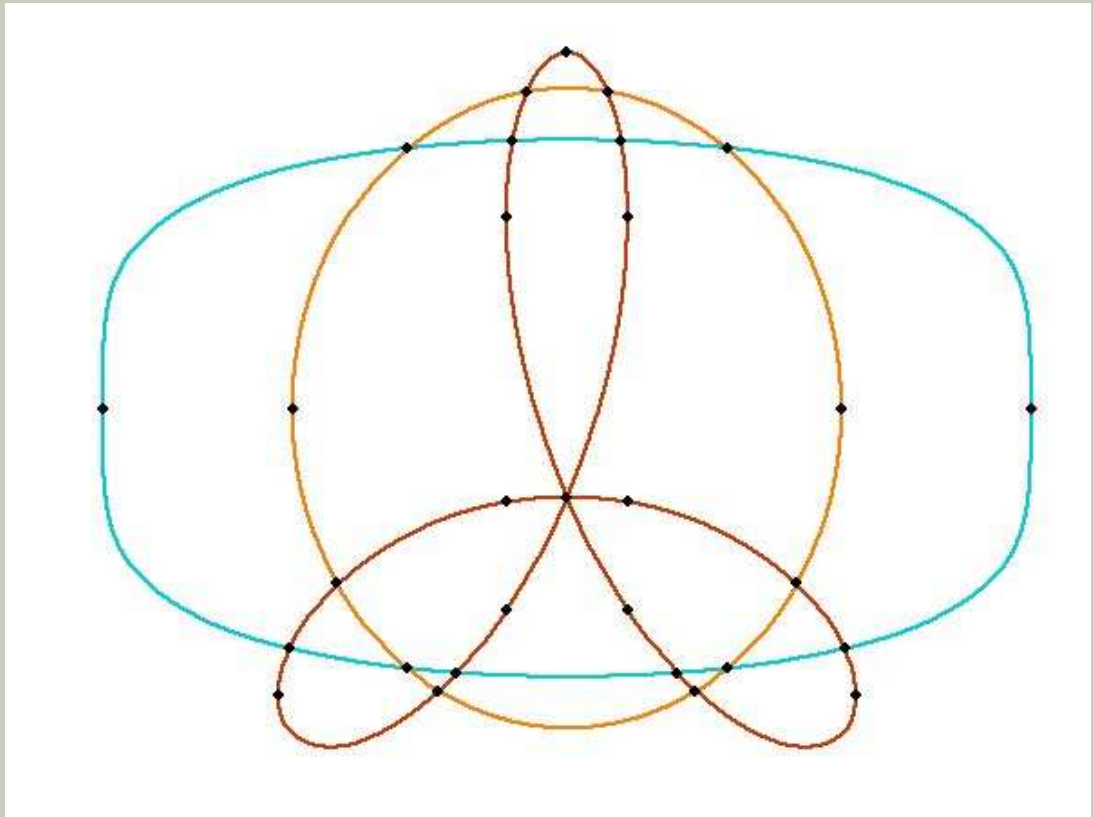
max planck institut informatik

- was assum
- uses CGA
  efficient b methods

Volpert 07/08]

- Curved_ke geometric
  operations
  eliyanenko 08]
- supports c

Example of three curves



Arrangements of algebraic curves

max planck institut informatik

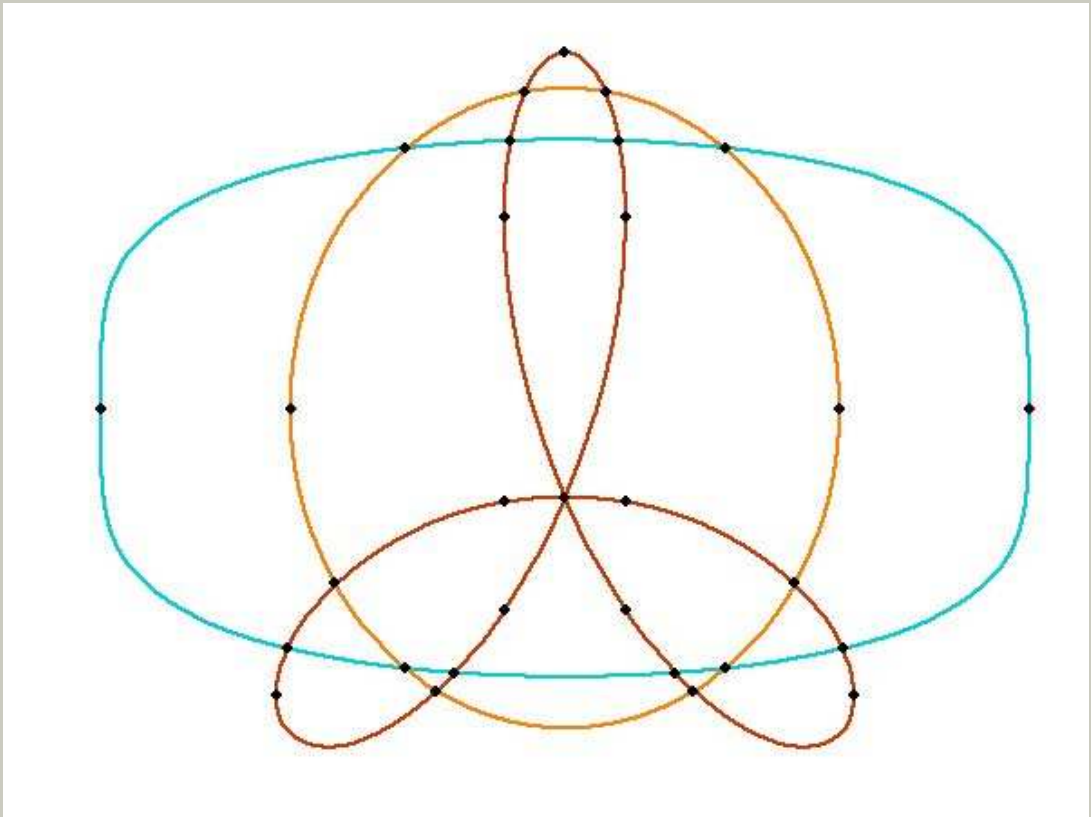
- was assum
- uses CGA
  efficient b methods

Volpert 07/08]

- Curved_ke geometric
  operations
- supports c

eliyanenko 08]

Example of three curves

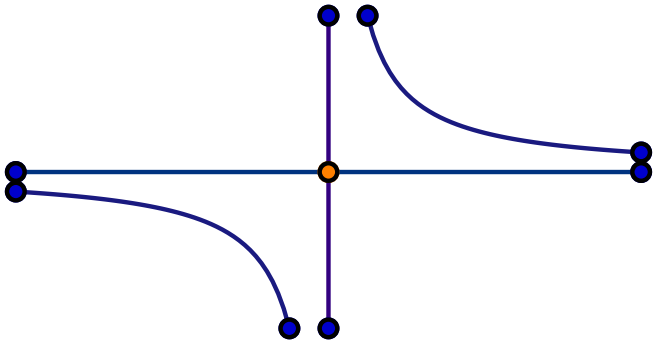


But: Most algebraic curves are unbounded!

Unbounded plane

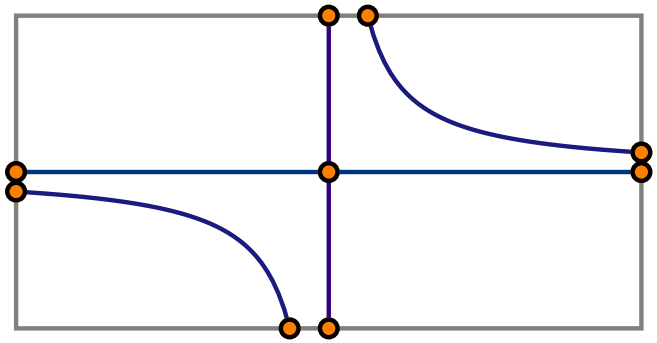
max planck institut informatik

Clipping the curves



- Sweep algorithm is unchanged
- Not online
- Single unbounded face

Infimaximal box [Mehlhorn & Seel, 2003]



- Sweep line modifications for linear objects, larger bit-lengths
- Online (no clipping)
- Result has multiple unbounded faces (and one ficticious)

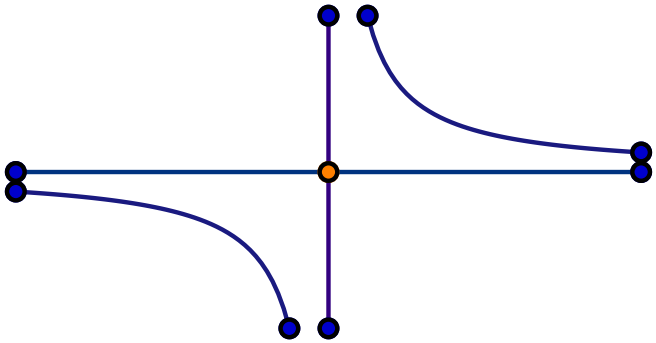
irst idea: GeometryTraits allows points at infinity

roblem 1: Must be implemented in each GeometryTraits of

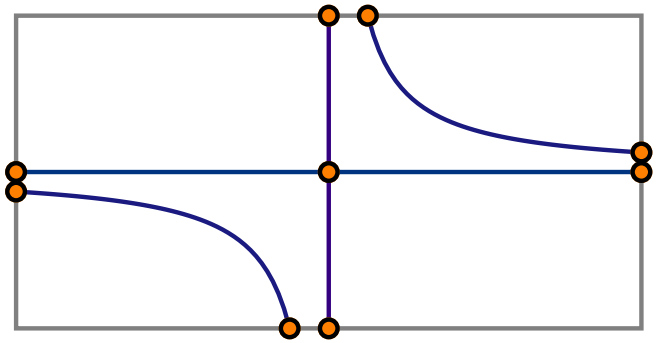
Unbounded plane

max planck institut informatik

Clipping the curves



Infimaximal box [Mehlhorn & Seel, 2003]

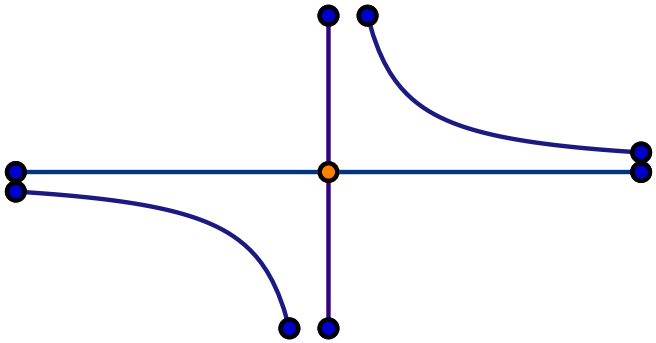


- First idea: GeometryTraits allows points at infinity
- Problem 1: Must be implemented in each GeometryTraits of unbounded curves
- Problem 2: Requires post-processing for unbounded faces
- New framework implements the usual duplicated parts
  - Demands a small set of simple functors
  - The programer is guided

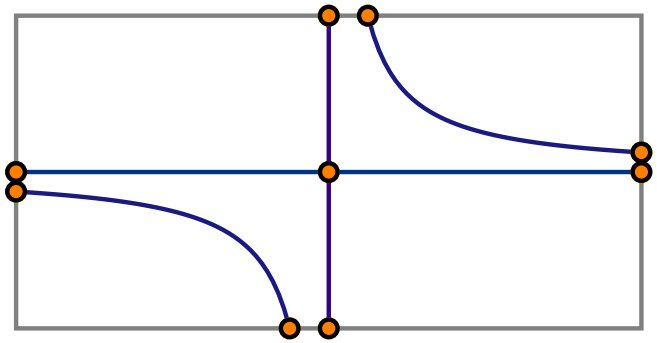
Unbounded plane

max planck institut informatik

Clipping the curves



Infimaximal box [Mehlhorn & Seel, 2003]



- First idea: GeometryTraits allows points at infinity
- Problem 1: Must be implemented in each GeometryTraits of unbounded curves
- Problem 2: Requires post-processing for unbounded faces
- New framework implements the usual duplicated parts
  - Demands a small set of simple functors
  - The programer is guided

Generalization: Parametric surface

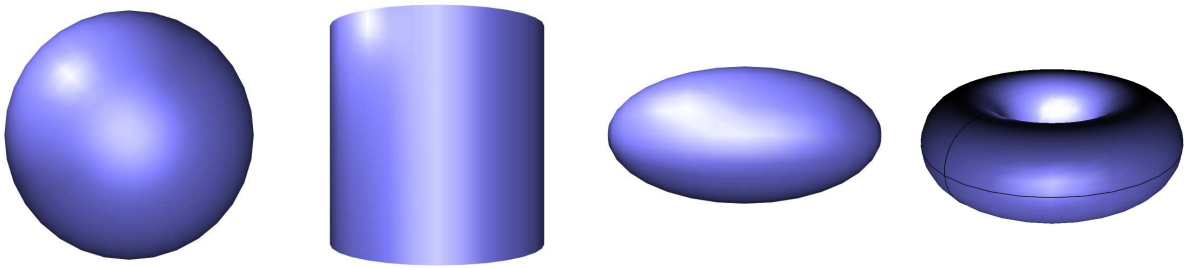
max planck institut informatik

Definition (Parametric surface)

An orientable *parametric surface* P in u and v is defined by

$$\phi_P(u, v) = (x(u, v), y(u, v), z(u, v))$$

where $\phi_P : \Phi \rightarrow \mathbb{R}^3$ and $P = \phi_P(\Phi)$, with $\Phi = UxV$ being a continuous and simply connected two-dimensional parameter space.



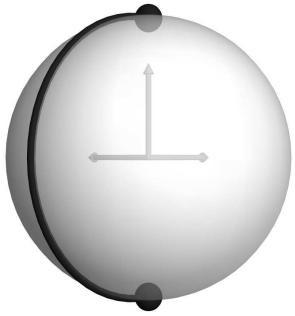
Sphere: $\phi_{sp}(u, v) = (r \cos u \cos v, r \sin u \cos v, r \sin v)$

$u \in [-\pi, \pi), v \in [-\frac{\pi}{2}, \frac{\pi}{2}]$

On a surface?!

max planck institut informatik

- Given: Parametric surface P; set of curves $\mathcal{C}$ embedded in P
- Goal: Compute and maintain arrangement



Motivation and Problems:

- P may be unbounded, also the curves in $\mathcal{C}$
  - Minimization diagram of unbounded surfaces
  - Stratification of surfaces
- "international date line", poles, ...

Solution in CGAL

Extended CGAL's `Arrangement_2` package [TAU]

- Unified software framework for parametric surfaces: `Arrangement_on_surface_2`
- Support for curves for ring Dupin cyclides cyclides (and quadrics)

On a surface?!

max planck institut informatik

- Given: Parametric surface P; set of curves $\mathcal{C}$ embedded in P
- Goal: Compute and maintain arrangement

Motivation and Problems:

- P may be unbounded, also the curves in $\mathcal{C}$
  - Minimization diagram of unbounded surfaces
  - Stratification of surfaces
- "international date line", poles, ...

Solution in CGAL

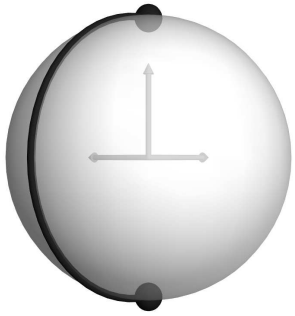
Extended CGAL's `Arrangement_2` package [TAU]

- Unified software framework for parametric surfaces: `Arrangement_on_surface_2`
- Support for curves for ring Dupin cyclides cyclides (and quadrics)

On a surface?!

max planck institut informatik

- Given: Parametric surface P; set of curves $\mathcal{C}$ embedded in P
- Goal: Compute and maintain arrangement



Motivation and Problems:

- P may be unbounded, also the curves in $\mathcal{C}$
  - Minimization diagram of unbounded surfaces
  - Stratification of surfaces
- "international date line", poles, ...

Solution in CGAL

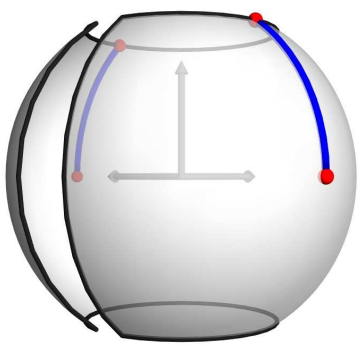
Extended CGAL's `Arrangement_2` package [TAU]

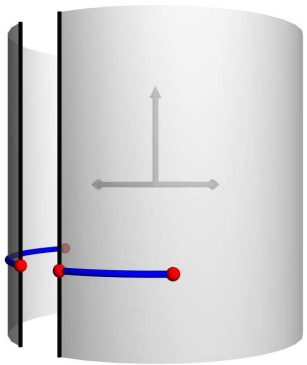
- Unified software framework for parametric surfaces: `Arrangement_on_surface_2`
- Support for curves for ring Dupin cyclides cyclides (and quadrics)

Arrangement on a surface

max planck institut informatik

- Decompose Φ into 5 parts: left, right, bottom, top, and interior
- Classify boundaries of parameter space ($\partial\Phi$) — four options:
  - finite, or infinite, i.e., open end
  - contracted, e.g., $\forall u \in U, \phi_S(u, v_{\min}) = p_0 \in P$
  - identified, e.g., $\forall v \in V, \phi_S(u_{\min}, v) = \phi_S(u_{\max}, v)$
- Split the input curves into u-monotone sweepable subcurves that are *interior disjoint* from the boundaries (see figures)
- Categorize each curve-end of subcurve according to its position

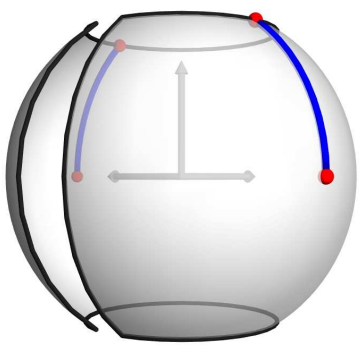


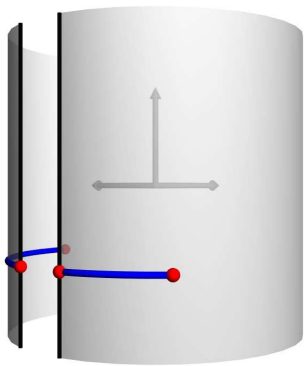


Arrangement on a surface

max planck institut informatik

- Decompose Φ into 5 parts: left, right, bottom, top, and interior
- Classify boundaries of parameter space ($\partial\Phi$) — four options:
  - finite, or infinite, i.e., open end
  - contracted, e.g., $\forall u \in U, \phi_S(u, v_{\min}) = p_0 \in P$
  - identified, e.g., $\forall v \in V, \phi_S(u_{\min}, v) = \phi_S(u_{\max}, v)$
- Split the input curves into u-monotone sweepable subcurves that are *interior disjoint* from the boundaries (see figures)
- Categorize each curve-end of subcurve according to its position





No curve-end on boundary

⇒ Situation is "isomorphic" to bounded curves in the plane

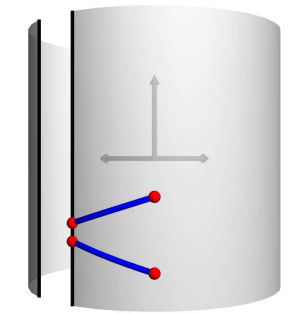
⇒ special handling only if curves touching the boundary

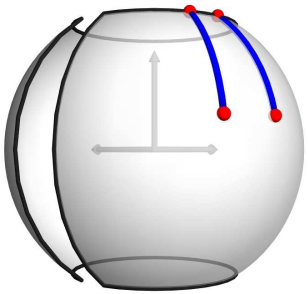
Problem 1: Geometry

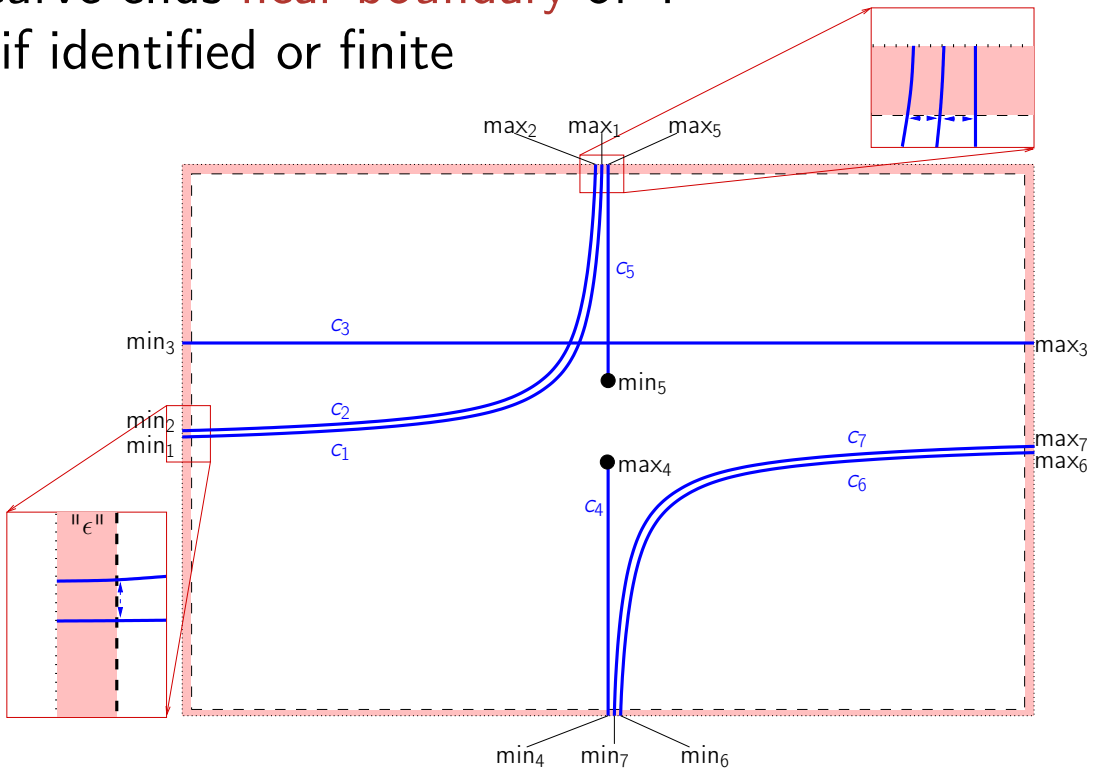
max planck institut informatik

Task: Interface lexicographic order of *points* and *curve-ends* in Φ

- Augment *ArrangementTraits_2* with surface-specific set of
  - simple comparisons of curve-ends near boundary of Φ
  - and on boundary of Φ, if identified or finite







Problem 1: Geometry

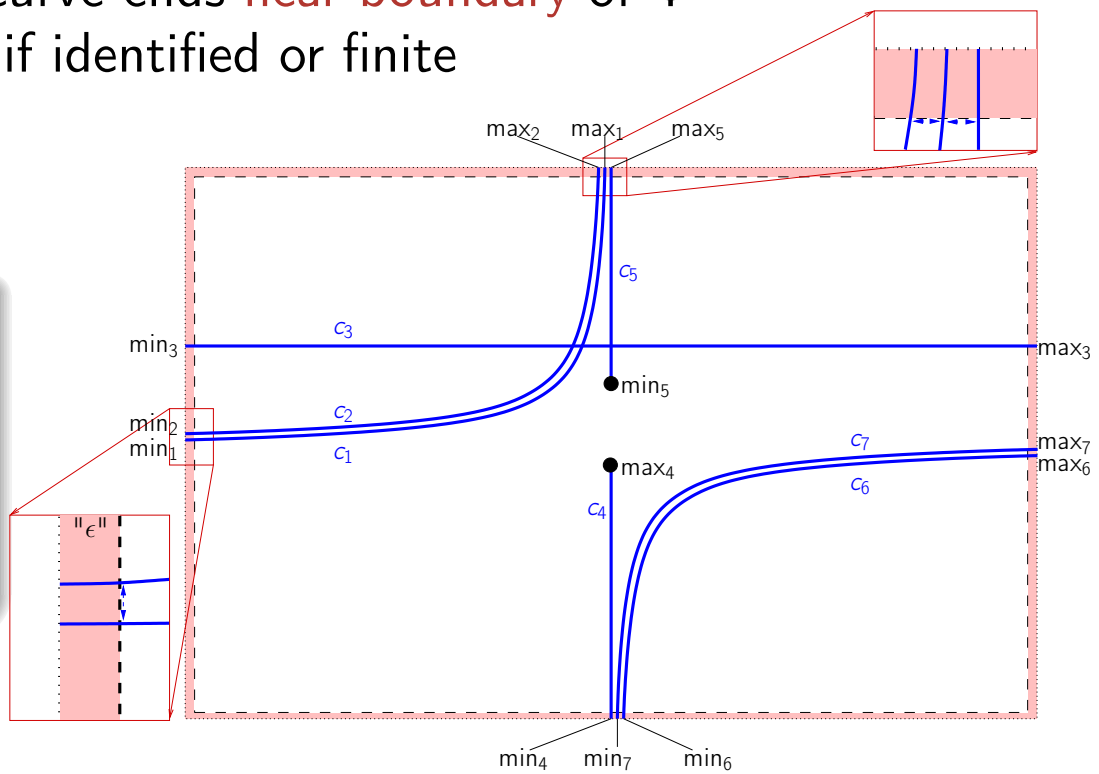
max planck institut informatik

Task: Interface lexicographic order of *points* and *curve-ends* in Φ

- Augment *ArrangementTraits_2* with surface-specific set of
  - simple comparisons of curve-ends near boundary of Φ
  - and on boundary of Φ, if identified or finite

Infinite algebraic curve-ends

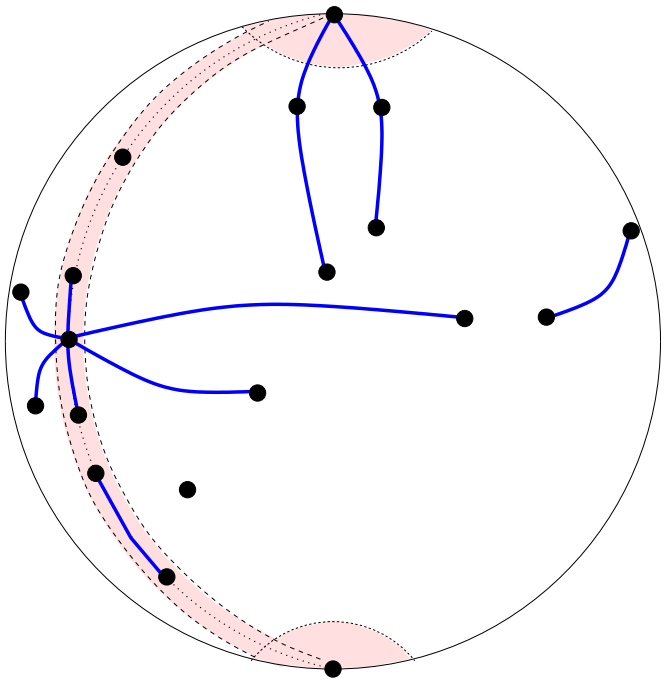
- compare to the right of $x = -\infty$
- compare to the left of $x = +\infty$
- compute how two approach a vertical asymptote

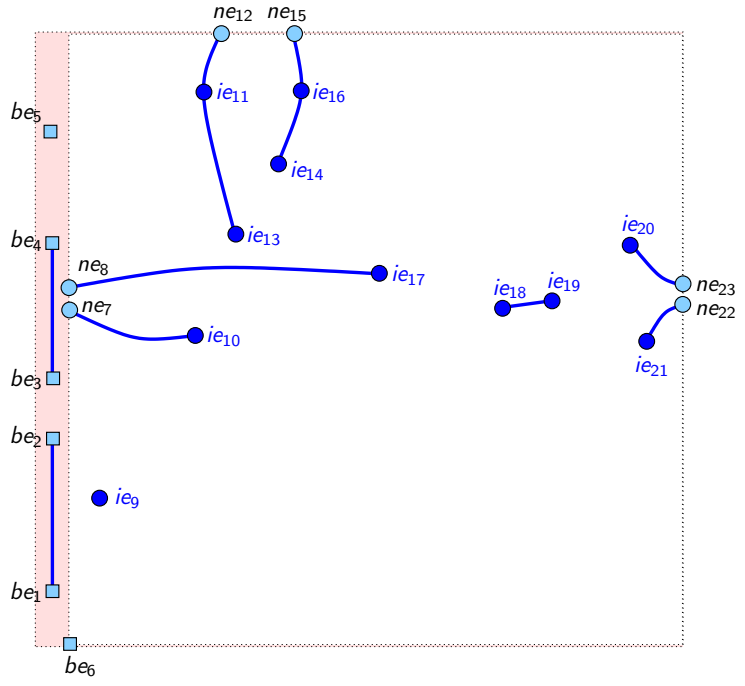


Lexicographic order in Φ

max planck institut informatik

- Package provides case distinction combining them
- No duplicated (surface-independent) code in each geometry traits
- Example: Lexicographic order in Φ for curves/points on sphere



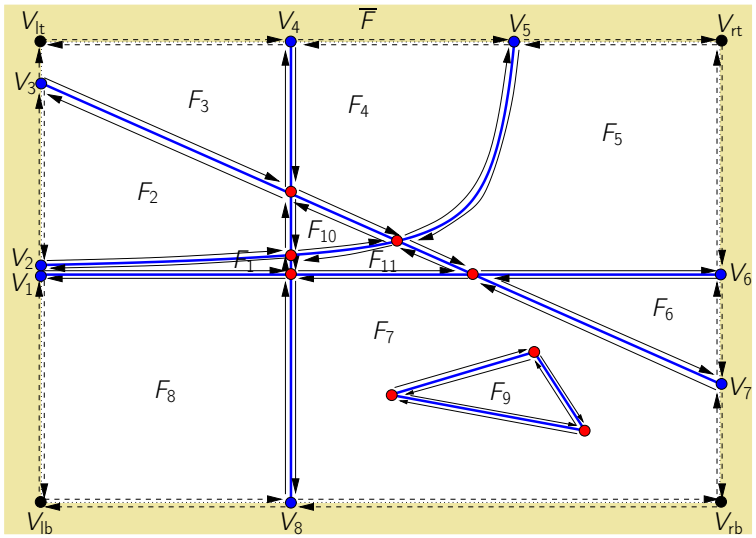


- Output of predicates is wrt Φ - **not their implemention**

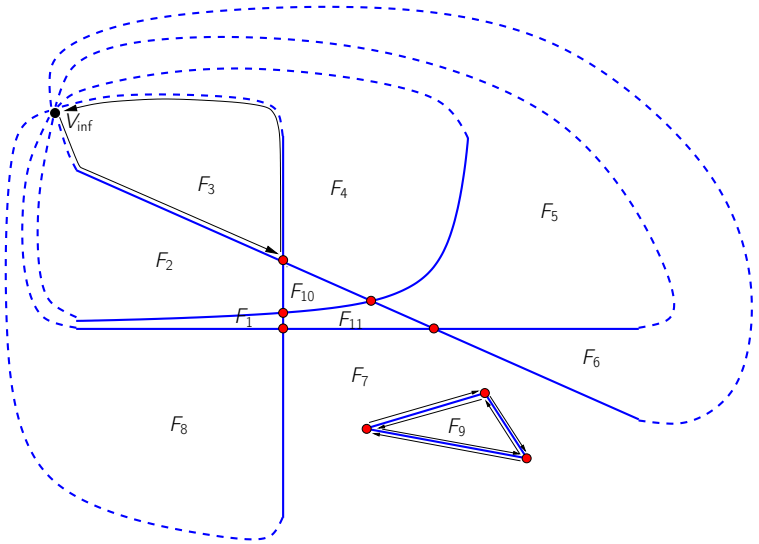
Problem 2: DCEL

max planck institut informatik

- Unique in planar case
- Already in unbounded plane different possibilities. Two examples:



bounding fictitious rectangle



single fictitious vertex

Task: Maintain DCEL-records related to $\partial\Phi$ respecting topology of P

New parameter: TopologyTraits

max planck institut informatik

```
template <typename GeoTraits, typename TopTraits>
class Arrangement_on_surface_2
```

Helps to *determine* $\mathcal{A}(\mathcal{C})$'s actual representation:

- Locate and maintain vertices on boundary
- Locate curves incident to such vertices
- Construct and maintain possible fictitious edges
- Help to decide whether insertion of curve results in a face splitting
- Assign boundary cycles to maintain consistent nesting graph
  - Tree strategy: There is always an "outermost" root face
  - Forest strategy: Maintain (several) equitable outermost faces

One TopologyTraits fits for all geometries on that surface.

New parameter: TopologyTraits

max planck institut informatik

```
template <typename GeoTraits, typename TopTraits>
class Arrangement_on_surface_2
```

Helps to *determine* $\mathcal{A}(\mathcal{C})$'s actual representation:

- Locate and maintain vertices on boundary
- Locate curves incident to such vertices
- Construct and maintain possible fictitious edges
- Help to decide whether insertion of curve results in a face splitting
- Assign boundary cycles to maintain consistent nesting graph
  - Tree strategy: There is always an "outermost" root face
  - Forest strategy: Maintain (several) equitable outermost faces

One TopologyTraits fits for all geometries on that surface.

New parameter: TopologyTraits

max planck institut informatik

```
template <typename GeoTraits, typename TopTraits>
class Arrangement_on_surface_2
```

Helps to *determine* $\mathcal{A}(\mathcal{C})$'s actual representation:

- Locate and maintain vertices on boundary
- Locate curves incident to such vertices
- Construct and maintain possible fictitious edges
- Help to decide whether insertion of curve results in a face splitting
- Assign boundary cycles to maintain consistent nesting graph
  - Tree strategy: There is always an "outermost" root face
  - Forest strategy: Maintain (several) equitable outermost faces

One TopologyTraits fits for all geometries on that surface.

New parameter: TopologyTraits

max planck institut informatik

```
template <typename GeoTraits, typename TopTraits>
class Arrangement_on_surface_2
```

Helps to *determine* $\mathcal{A}(\mathcal{C})$'s actual representation:

- Locate and maintain vertices on boundary
- Locate curves incident to such vertices
- Construct and maintain possible fictitious edges
- Help to decide whether insertion of curve results in a face splitting
- Assign boundary cycles to maintain consistent nesting graph
  - Tree strategy: There is always an "outermost" root face
  - Forest strategy: Maintain (several) equitable outermost faces

One TopologyTraits fits for all geometries on that surface.

New parameter: TopologyTraits

max planck institut informatik

```
template <typename GeoTraits, typename TopTraits>
class Arrangement_on_surface_2
```

Helps to *determine* $\mathcal{A}(\mathcal{C})$'s actual representation:

- Locate and maintain vertices on boundary
- Locate curves incident to such vertices
- Construct and maintain possible fictitious edges
- Help to decide whether insertion of curve results in a face splitting
- Assign boundary cycles to maintain consistent nesting graph
  - Tree strategy: There is always an "outermost" root face
  - Forest strategy: Maintain (several) equitable outermost faces

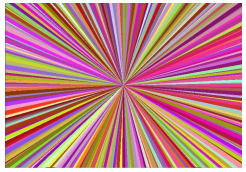
One TopologyTraits fits for all geometries on that surface.

Available TopologyTraits classes

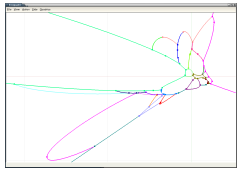
max planck institut informatik

- Bounded plane (original case)
- Unbounded plane

Arrangements of algebraic curves



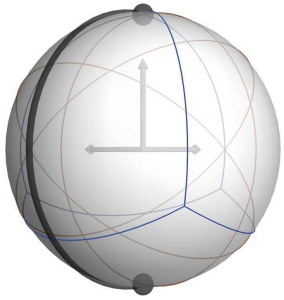
Voronoi diagrams of points in the plane



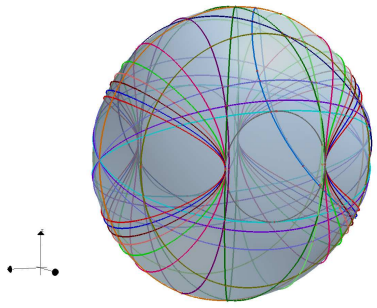
Lower envelopes of quadrics

- Surfaces

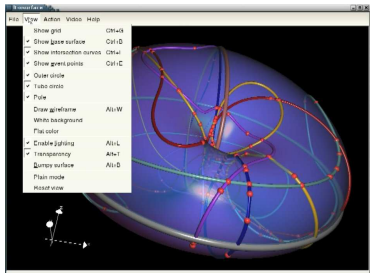
Sphere



Quadrics



Dupin cyclides



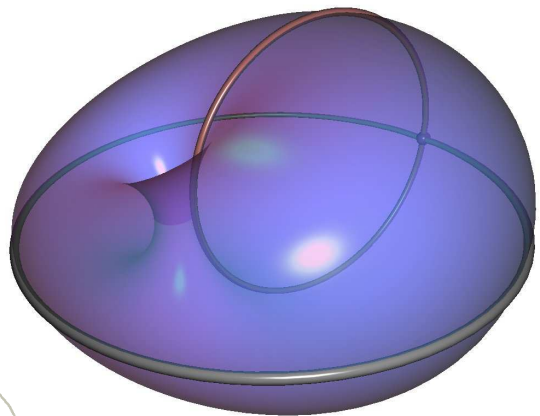
Reference Cyclide

max planck institut informatik

- Cyclide: Surface homeomorphic to torus
- Standard algebraic parameterization Φ for cyclide P (no sin/cos!)

$$\begin{pmatrix} u \\ v \end{pmatrix} \mapsto \begin{pmatrix} \frac{\mu(c(1+u^2)(1+v^2)-a(1-v^2)(1-u^2))+b^2(1-u^2)(1+v^2)}{a(1+u^2)(1+v^2)-c(1-u^2)(1-v^2)} \\ \frac{2u(a(1+v^2)-\mu(1-v^2))b}{a(1+u^2)(1+v^2)-c(1-u^2)(1-v^2)} \\ \frac{2v(c(1-u^2)-\mu(1+u^2))b}{a(1+u^2)(1+v^2)-c(1-u^2)(1-v^2)} \end{pmatrix}$$

- Splits cyclide at *cut circles* and induces *pole* ("unfolding to the real plane")



- Also given: Set of algebraic surfaces $\mathcal{S}$ intersecting P
- Goal: Arrangement induced by $S \cap P$ with all $S \in \mathcal{S}$

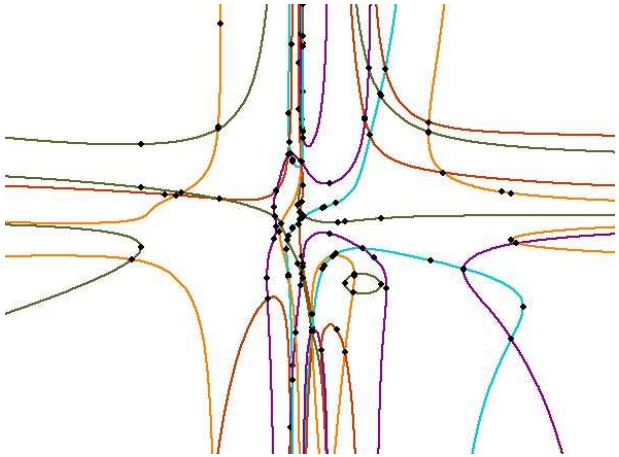
Video [GfX: Pavel Emeliyanenko]

GeometryTraits: Algebraic curves

max planck institut informatik

- Surface S defined by $f := f(x, y, z) \in \mathbb{Q}[x, y, z]$, $D := \deg(f)$
- $F(u, v) := f(x(u, v), y(u, v), z(u, v)) \in \mathbb{Q}[u, v]$
- defines real algebraic plane curve of bidegree $(2D, 2D)$

Curves in Φ_P induced by 5 surfaces of degree 3



Enhance planar geometry for cyclide

- Interpret arcs towards infinity
  $\Rightarrow$ arcs towards cut circles
- Interpret comparisons near infinity
  $\Rightarrow$ comparisons near cut circles
- New: Comparisons on the boundary

Cyclidean TopologyTraits

max planck institut informatik

- **Problem:** Cut circles have two (four) preimages in Φ

  Goal: Store one DCEL-vertex for each set of identified preimages

  Solution: Two sorted sequences of vertices + one for pole
  - coordinates in Φ are given by asymptotes of curves in Φ
  - Task: Assign "unbounded" curve-arcs to *asymptotes*
  - Vertical: (f, x, a), but non-vertical: $(f, \pm\infty, a)$

    $\Rightarrow$ Buckets; *one* for each asymptote for $x \to -\infty$
  - get *safe* u_0: real roots of $F(u_0, v)$ do not leave bucket for $u < u_0$
- **Problem:** Adding curve that closes cycle $\Rightarrow$ new face?
  - first red curve $\Rightarrow$ **NO new face**

    this cycle is non-seperating
  - second red curve $\Rightarrow$ **NEW face**

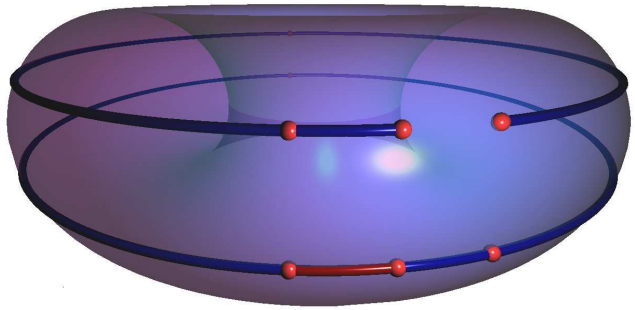
    this cycle is seperating
  - Boundary cycles: **Forest-strategy**

    face that contains non-contractible cycle $\Rightarrow$ root in nesting graph

Cyclidean TopologyTraits

max planck institut informatik

- **Problem:** Cut circles have two (four) preimages in Φ
  Goal: Store one DCEL-vertex for each set of identified preimages
  Solution: Two sorted sequences of vertices + one for pole
  - coordinates in Φ are given by asymptotes of curves in Φ
  - Task: Assign "unbounded" curve-arcs to *asymptotes*
  - Vertical: (f, x, a), but non-vertical: $(f, \pm\infty, a)$
    $\Rightarrow$ Buckets; *one* for each asymptote for $x \to -\infty$
  - get *safe* u_0: real roots of $F(u_0, v)$ do not leave bucket for $u < u_0$
- **Problem:** Adding curve that closes cycle $\Rightarrow$ new face?
  - first red curve $\Rightarrow$ **NO new face**
    this cycle is non-seperating
  - second red curve $\Rightarrow$ **NEW face**
    this cycle is seperating

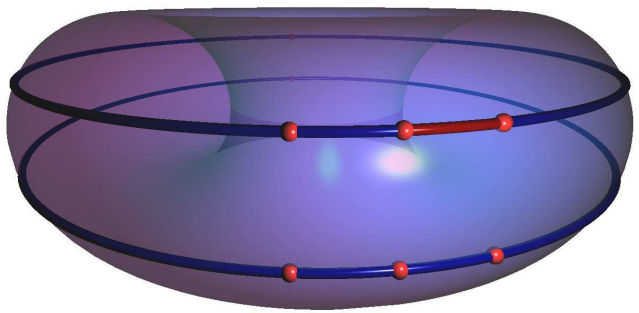


  - Boundary cycles: **Forest-strategy**
    face that contains non-contractible cycle $\Rightarrow$ root in nesting graph

Cyclidean TopologyTraits

max planck institut informatik

- **Problem:** Cut circles have two (four) preimages in Φ
  Goal: Store one DCEL-vertex for each set of identified preimages
  Solution: Two sorted sequences of vertices + one for pole
  - coordinates in Φ are given by asymptotes of curves in Φ
  - Task: Assign "unbounded" curve-arcs to *asymptotes*
  - Vertical: (f, x, a), but non-vertical: $(f, \pm\infty, a)$
    $\Rightarrow$ Buckets; *one* for each asymptote for $x \to -\infty$
  - get *safe* u_0: real roots of $F(u_0, v)$ do not leave bucket for $u < u_0$
- **Problem:** Adding curve that closes cycle $\Rightarrow$ new face?
  - first red curve $\Rightarrow$ **NO new face**
    this cycle is non-seperating
  - second red curve $\Rightarrow$ **NEW face**
    this cycle is seperating



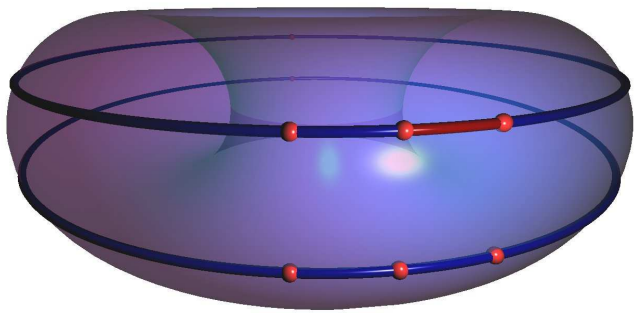
  - Boundary cycles: **Forest-strategy**
    face that contains non-contractible cycle $\Rightarrow$ root in nesting graph

Cyclidean TopologyTraits

max planck institut informatik

- **Problem:** Cut circles have two (four) preimages in Φ
  Goal: Store one DCEL-vertex for each set of identified preimages
  Solution: Two sorted sequences of vertices + one for pole
  - coordinates in Φ are given by asymptotes of curves in Φ
  - Task: Assign "unbounded" curve-arcs to *asymptotes*
  - Vertical: (f, x, a), but non-vertical: $(f, \pm\infty, a)$
    $\Rightarrow$ Buckets; *one* for each asymptote for $x \to -\infty$
  - get *safe* u_0: real roots of $F(u_0, v)$ do not leave bucket for $u < u_0$

- **Problem:** Adding curve that closes cycle $\Rightarrow$ new face?
  - first red curve $\Rightarrow$ **NO new face**
    this cycle is non-seperating
  - second red curve $\Rightarrow$ **NEW face**
    this cycle is seperating



  - Boundary cycles: **Forest-strategy**
    face that contains non-contractible cycle $\Rightarrow$ root in nesting graph

Experiments: Constructions

max planck institut informatik

- Created interpolated surfaces (some with degeneracies wrt cyclide)
- Constructed arrangement on cyclide with sweep line (C)
- Constructed arrangement in unbounded plane (2D)

Instance-<deg>	#S	#V,#E,#F (C)	t in s (C)	t in s (2D)
ipl-1	10	119,190,71	**0.14**	0.14
ipl-1	20	384,682, 298	**0.58**	0.58
ipl-1	50	1837,3363,1526	**2.14**	2.00
ipl-2	10	358,575,217	**1.07**	1.25
ipl-2	20	1211,2147,937	**3.14**	3.04
ipl-3	10	542,847,305	**4.84**	4.62
ipl-3-6points	10	680,1092,412	**32.43**	31.17
ipl-3-2sing	10	694,1062,368	**5.82**	5.57
ipl-4	10	785,1204,419	**50.42**	49.97
ipl-4-6points	10	989,1529,540	**461.74**	450.54
ipl-4-2sing	10	933,1471,538	**53.01**	52.78

- Choice of TopologyTraits does not influence performance
- Running time spent in analyses of planar curves

Moebius strip

max planck institut informatik

- is also rational parameterizable (as the cyclide is)
- but arrangement on this NON-ORIENTABLE surfaces cannot be represented as DCEL.
- requires quad-edge data structure [Guibas and Stolfi]
- not (yet) available in CGAL

Applications

max planck institut informatik

- Adjacency graph of surfaces: [abstract version of Hemmer 08]
  Identify geometrically equal vertices on different surfaces
- Enhance data-structure to model arrangement **of** surfaces
- Arrangements on spheres: Minkowsi-Sums of (convex) polyhedra, assembly planning of polyhedra [Fogel 08]
- Configuration space of (some) two-link robot arms moving respecting obstacles: arrangement on a torus. Example: Molecules of some amino-acids have two main rotation axes

Robot Motion Planning

max planck institut informatik

- Goal: giving a motion description for a collision-"free" movement of a "robot" respecting obstacles
- also known as the 'Piano Mover's Problem'

Robot

max planck institut informatik

- rather general term
- not needed to be motorized ("Piano")
- has some shape (of finite description)
- might be static, or has some degrees of freedom (arms)
- typical Movements: translation, rotation
- typical robots: objects, biological molecules, robotic manipulators, animated digital characters

Collision-free movement

max planck institut informatik

- Robot/object is allowed to freely move (according to its possibilites) in some domain, e.g., a room/flat.
- motion must be continuous
- Domain might be restricted by obstacles: Walls, furniture
- Two choices:
  - totally collision-free
  - "sliding" along an obstacle is allowed

Geometry

max planck institut informatik

- Workspace for domain, robot, and obstacles is 2D or 3D Euclidean space
- Motion is given as a (one-dimensional) *path* in *configuration space* C
- C might have much higher dimension (also depends on robot's number of DOF)

C-Space (Examples)

max planck institut informatik

Robot is single point

- Translating the robot in a 2D workspace, C is given by the position (x, y) in the plane: $\dim C = 2$.

Robot is 2D shape in a 2D workspace

- Translation: Take a reference point (e.g., a corner). Robot's position is specified by position of reference (x, y): $\dim C = 2$.
- With rotation: Add angle-parameter $\phi \in [0, 2\pi)$. Robot's configuration is given by triple: (x, y, ϕ) in $\mathbb{R}^2 \times [0, 2\pi)$: $\dim C = 3$

Robot is a 3D shape in 3D workspace

- Reference point (x, y, z) plus three Euler angles (α, β, γ): $\dim C = 6$

C-Space (Examples)

max planck institut informatik

Robot is single point

- Translating the robot in a 2D workspace, C is given by the position (x, y) in the plane: $\dim C = 2$.

Robot is 2D shape in a 2D workspace

- Translation: Take a reference point (e.g., a corner). Robot's position is specified by position of reference (x, y): $\dim C = 2$.
- With rotation: Add angle-parameter $\phi \in [0, 2\pi)$. Robot's configuration is given by triple: (x, y, ϕ) in $\mathbb{R}^2 \times [0, 2\pi)$: $\dim C = 3$

Robot has DOF

- A revolute join (Drehgelenk) adds a dimension to C.

Robo

- Reference point (x, y, z) plus three Euler angles (α, β, γ): $\dim C = 6$

Free-space

max planck institut informatik

- The subset $C_{\text{free}} \subseteq C$ that avoids collisions with any obstacle (touching or penetrative) is called the *free-space* (of the robot)
- The complement of C_{free} is the *obstacle* or *forbidden region*

Algorithms

max planck institut informatik

General idea

- Locate initial position A in free-space
- Locate final position B in free-space
- Find continuous path connecting both positions
- If there is no such path: Motion from A to B not feasible

Algorithms - Grid based

max planck institut informatik

Work well for low dimensional C

- Overlay C with a grid (graph)
- Remove vertices and edges not fully contained in C_{free}
- Search shortest path in remaining graph between start and end configuration

Remarks

- Requires dense grid to find narrow passages, becoming slow
- Requires exponential number of vertices (in dim C)

Algorithms - Sample based

max planck institut informatik

Develop roadmap(-graph) in C_{free}.

- Sample n configurations in C; keep those in C_{free} as *milestones*
- Connect milestones P and Q with *road* (an edge) if $\overline{PQ} \subset C_{\text{free}}$
- Path-search adds A and B to roadmap:
  If connecting path can be found, return it. Else: "I don't know"

Remarks

- "State-of-the-art", even for high-dimensional C - though:
- Sampled milestones do not suffice to find connecting path
- Spending more time increases probability to find existing solution path towards 1
- Variations: test only neighbors, non-uniform sampling, quasirandom, tree-growing for few searches

Algorithms - Geometric approach

max planck institut informatik

They are complete, i.e.,

- always construct a feasible path if existing

Typical algorithm

- Construct C_{free}
- Decompose C_{free} into cells of "constant" size, e.g., by vertical decomposition
- Locate A and B in cells
- Use adjacency information of cells to conclude whether there is a free and continuous path passing cells and connecting A with B

Construction of C_{free}

max planck institut informatik

Minkowski sum

$$M = P \oplus Q = \{p + q \,|\, p \in P, q \in Q\}$$

Collision Detection

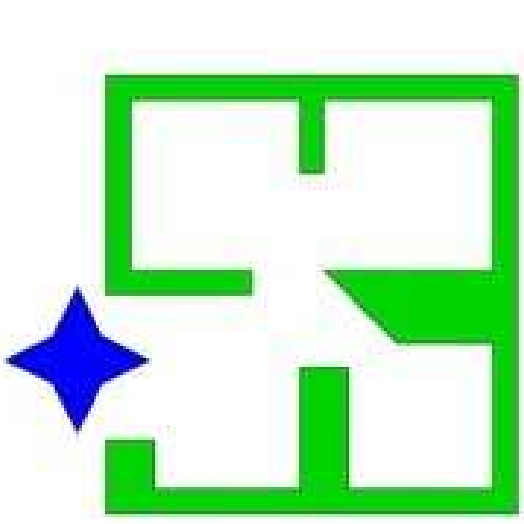
$$P \cap Q \neq \emptyset \Leftrightarrow 0 \in M' := P \oplus (-Q)$$

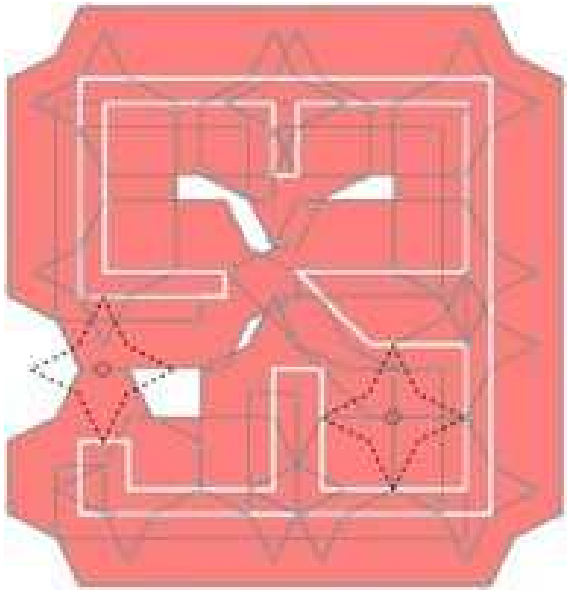
($-Q$ means inverting at the origin)

Computing C_{free}

- Consider the set of obstacles as P
- Consider the robot as Q
- $C_{\text{free}} := \overline{P \oplus (-Q)}$
- "Sliding the robot" along the obstacles

Examples





- Exact construction detects *one-dimensional* passage (sliding along at least two obstacles
- Rounded floating-point would probably be blind

Complexity

max planck institut informatik

P and Q convex polygons, n and m edges

- Space $O(m+n)$

Question: How to compute in linear time?

P and Q polygons, only one convex, n and m edges

- Space $O(nm)$

P and Q polygons, n and m edges

- Space $O((nm)^2)$

Remark: Minkowski sums are defined in any dimension. Computing is

Not covered

max planck institut informatik

- Rotations
- Analysis of DOFs

Voronoi diagram

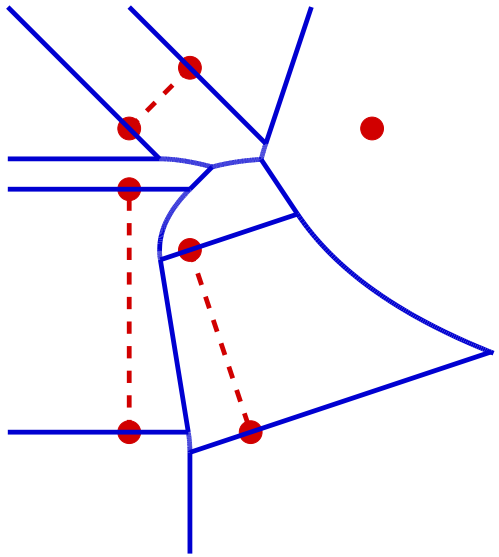
max planck institut informatik

Switch to slides by Ophir Setter, Tel-Aviv University.

Voronoi Diagrams

- Given *n* objects (Voronoi sites) in some space (e.g., $\mathbb{R}^d$, $\mathbb{S}^d$) and a distance function ρ
- The Voronoi Diagram subdivides the space into cells
- Each cell consists of points that are closer to one particular site than to any other site

- Variants include different:
  - Classes of sites
  - Embedding spaces
  - Distance functions (e.g., farthest-site Voronoi diagrams)

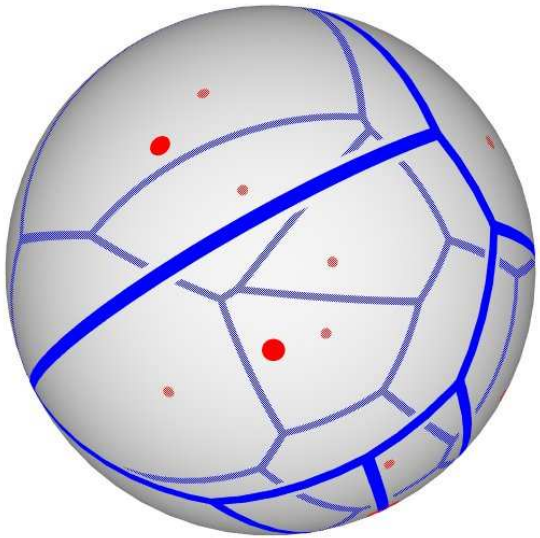


Voronoi diagram of segments and points

Voronoi Diagrams

- Given *n* objects (Voronoi sites) in some space (e.g., $\mathbb{R}^d$, $\mathbb{S}^d$) and a distance function ρ
- The Voronoi Diagram subdivides the space into cells
- Each cell consists of points that are closer to one particular site than to any other site

- Variants include different:
  - Classes of sites
  - Embedding spaces
  - Distance functions (e.g., farthest-site Voronoi diagrams)

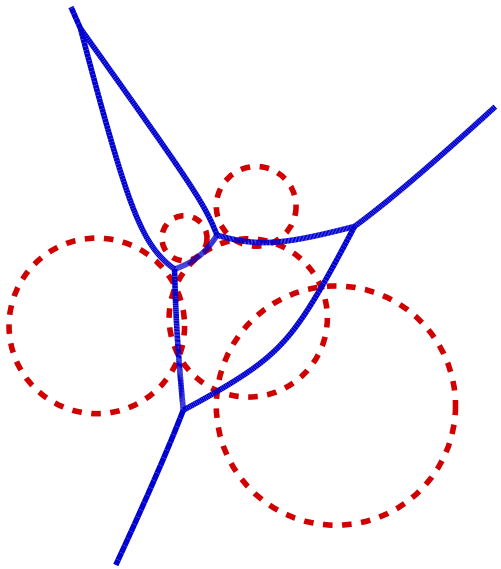


Voronoi diagram on the sphere

Voronoi Diagrams

- Given *n* objects (Voronoi sites) in some space (e.g., $\mathbb{R}^d$, $\mathbb{S}^d$) and a distance function ρ
- The Voronoi Diagram subdivides the space into cells
- Each cell consists of points that are closer to one particular site than to any other site

- Variants include different:
  - Classes of sites
  - Embedding spaces
  - Distance functions (e.g., farthest-site Voronoi diagrams)



Apollonius diagram
(additively-weighted Voronoi diagram)

Lower Envelopes

and Voronoi Diagrams

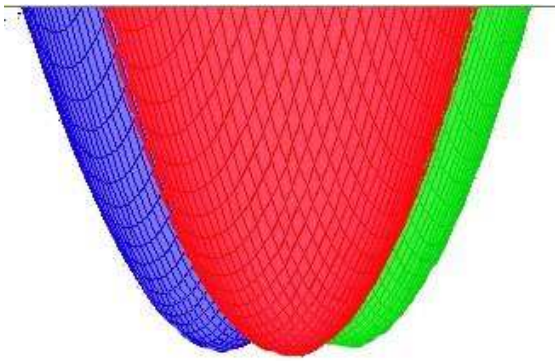
Definition

Given a set of bivariate functions $S = \{s_1, \ldots, s_n\}$, their lower envelope is defined to be their pointwise minimum:

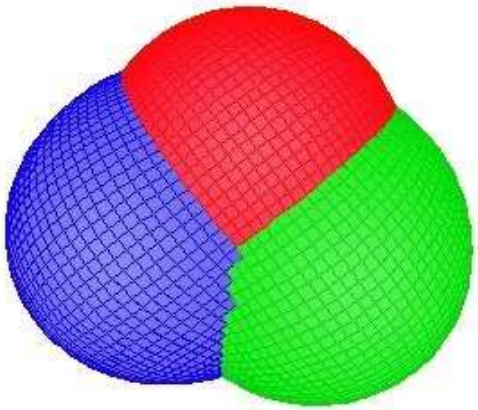
$$\Psi(x, y) = \min_{1 \leq i \leq n} s_i(x, y)$$

Corollary

Voronoi diagrams are the minimization diagrams of the distance functions from each site [Edelsbrunner & Seidel, 1986]



Distance functions are paraboloids

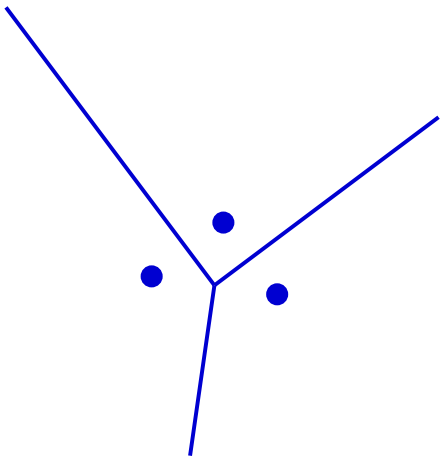


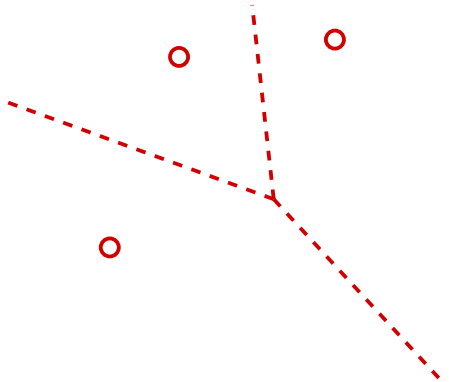
Looking from bottom gives us the Voronoi diagram

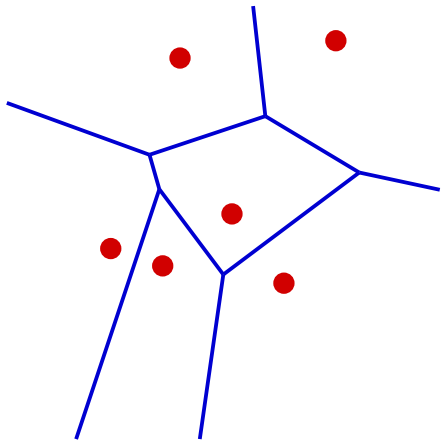
The Divide-and-Conquer Algorithm

Let S be a set of n sites

1. Partition S into two disjoint subsets S_1 and S_2 of equal size
2. Construct $\mathrm{Vor}_\rho(S_1)$ and $\mathrm{Vor}_\rho(S_2)$ recursively
3. Merge the two Voronoi diagrams to obtain $\mathrm{Vor}_\rho(S)$

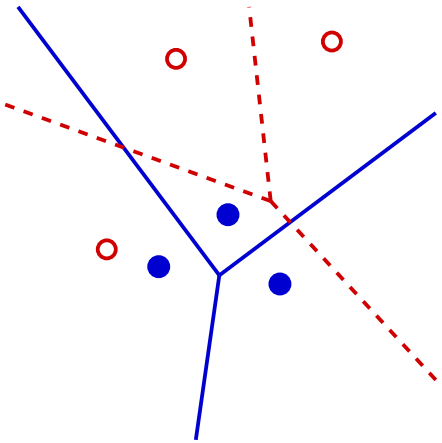






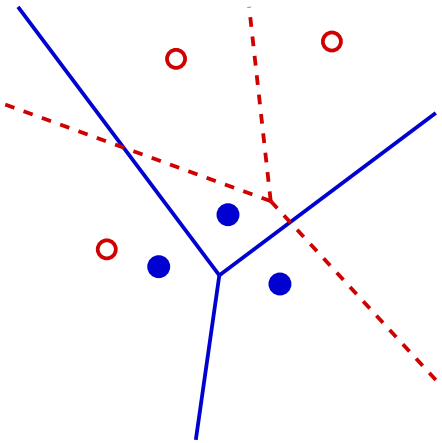
The Merging Step

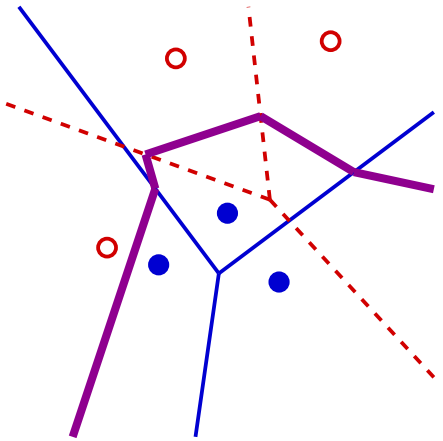
1. Overlay $\mathrm{Vor}_\rho(S_1)$ and $\mathrm{Vor}_\rho(S_2)$ using sweep



The Merging Step

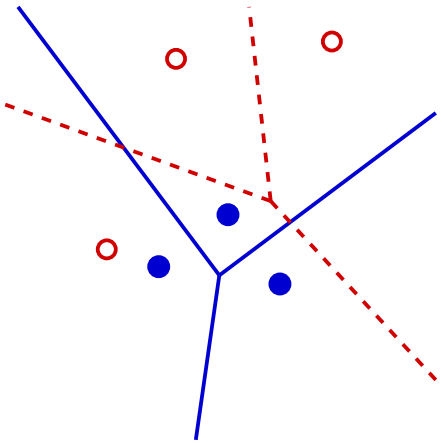
1. Overlay $\mathrm{Vor}_\rho(S_1)$ and $\mathrm{Vor}_\rho(S_2)$ using sweep
2. Partition each face to points closer to the site in S_1 and points closer to the site in S_2
3. Label feature of the refined overlay with the sites nearest to it

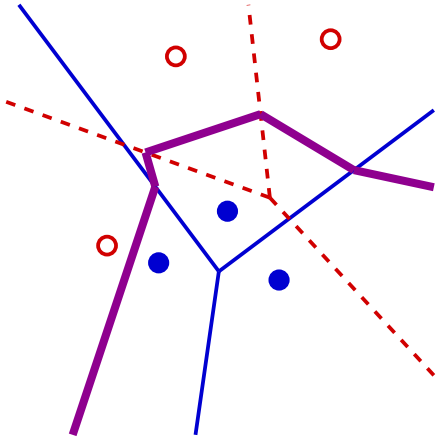


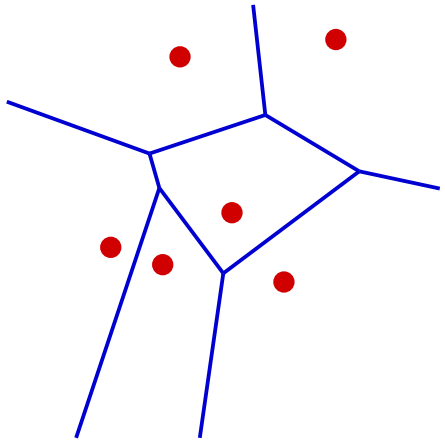


The Merging Step

1. Overlay $\mathrm{Vor}_\rho(S_1)$ and $\mathrm{Vor}_\rho(S_2)$ using sweep
2. Partition each face to points closer to the site in S_1 and points closer to the site in S_2
3. Label feature of the refined overlay with the sites nearest to it
4. Remove redundant features

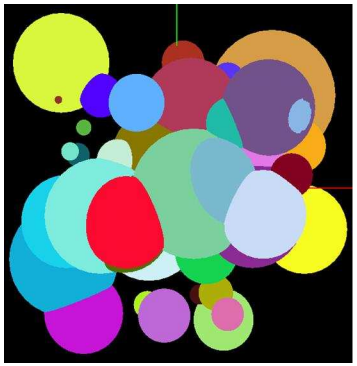
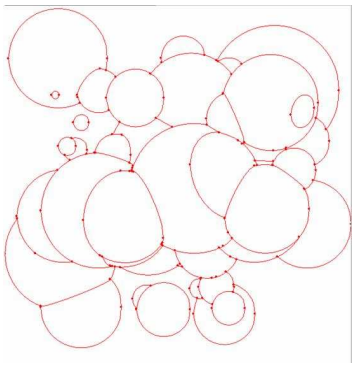
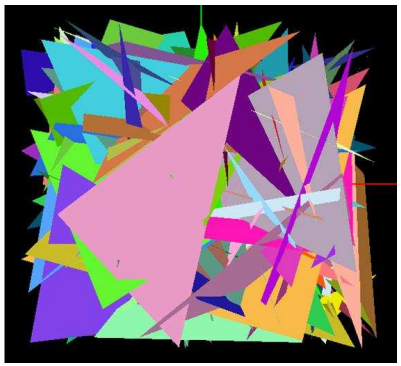
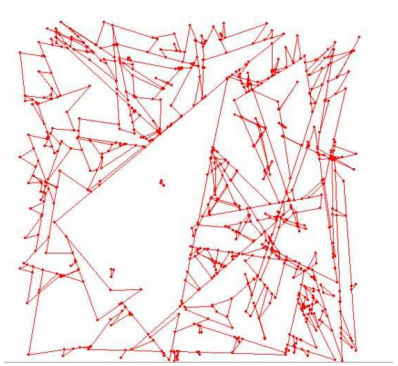






Envelopes and Arrangements in CGAL

- `Arrangement_on_surface_2` constructs and maintains arrangements on two-dimensional parametric surfaces
- `Envelope_3` package computes lower and upper envelopes of general surfaces [Meyerovitch, 2006]
- Robust and Exact
  - All inputs are handled correctly (including degenerate input)
  - Exact number types are used to achieve exact results
- Generic – Easy to interface, extend and adapt
- Modular – Geometric and topological aspects are separated

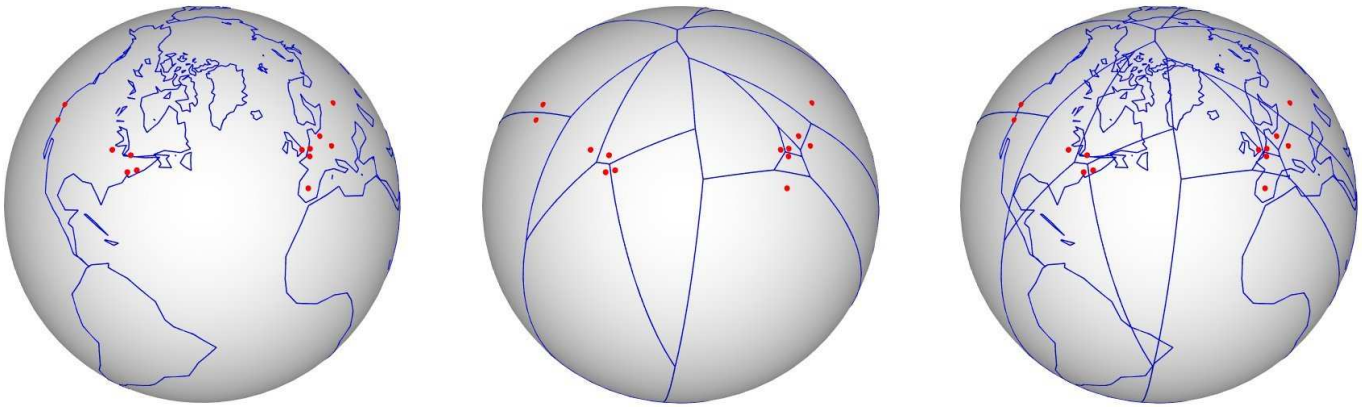
Implementation

- Reduced and simplified interface for diagrams with one-dimensional bisectors
- Computing diagrams, the bisector curves of which are currently supported by the arrangement package, is made easy (e.g., linear and circular arcs, algebraic curves, geodesics on the sphere)
- The framework supports types of diagrams that most frameworks do not support:
  - Quadratic-size diagrams, e.g., Möbius diagrams and triangle-area distance-function Voronoi diagrams
  - Non-connected bisectors, e.g., anisotropic Voronoi diagrams
  - Two-dimensional bisectors
- **Disadvantage:** Though general, the method uses exact constructions of bisectors and Voronoi vertices, which makes the running time inferior to various dedicated implementations (e.g., Delaunay triangulations in CGAL)

Other Advantages

The diagrams are represented as CGAL arrangements

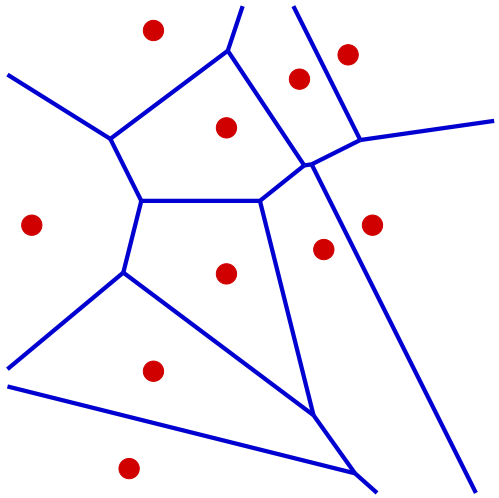
- The vertices, edges, and faces of the diagrams can easily be traversed while obtaining coordinates to any desired precision
- Point-location functionality
- Inserting and removing curves
- Overlay between diagrams, which is used, for example, for computing minimum-width annulus and for representing the local zones of two competing telecommunication operators [Baccelli et al., 2000]
- etc.



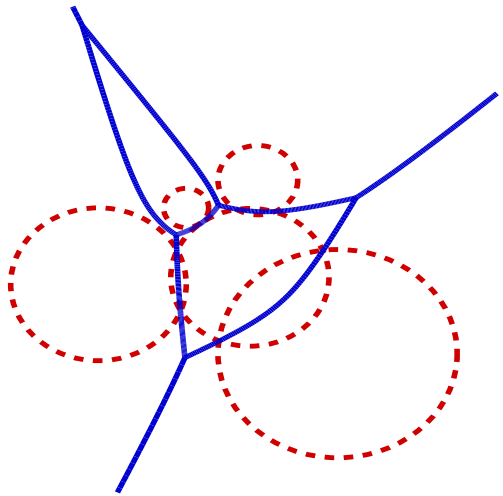
Overlaying an arrangement and a Voronoi diagram on the sphere

Examples of Available Diagrams

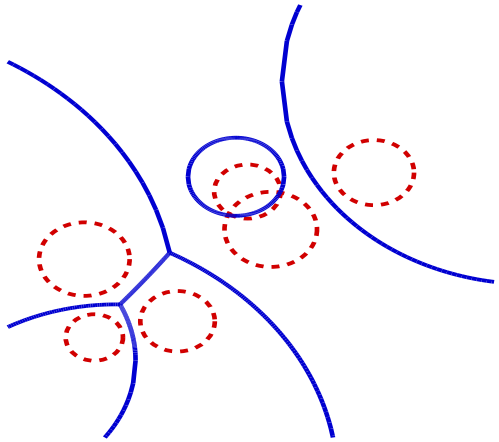
Nearest-Site Voronoi Diagrams



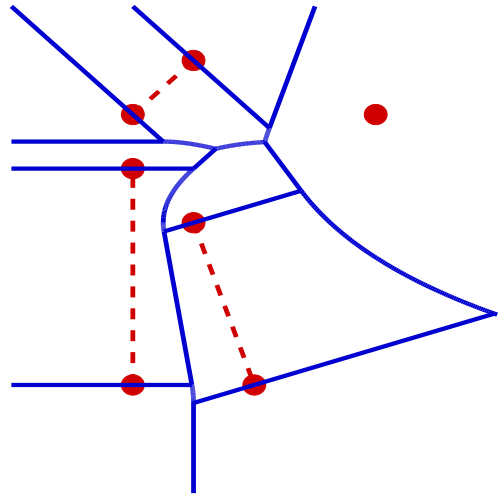
Standard Voronoi diagrams and power diagrams



Apollonius (additively-weighted Voronoi) diagrams



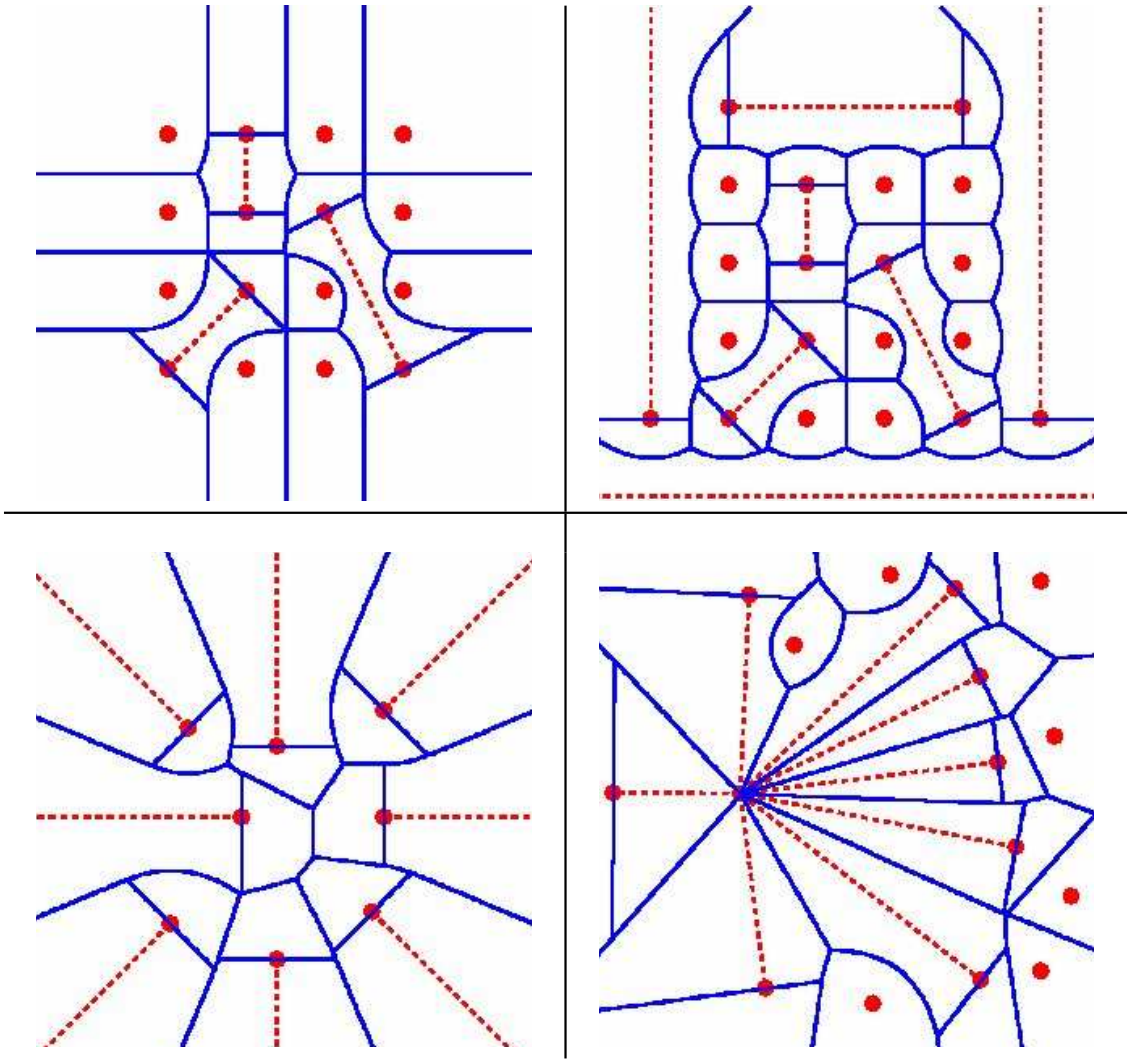
Möbius and anisotropic diagrams



Voronoi diagram of linear objects

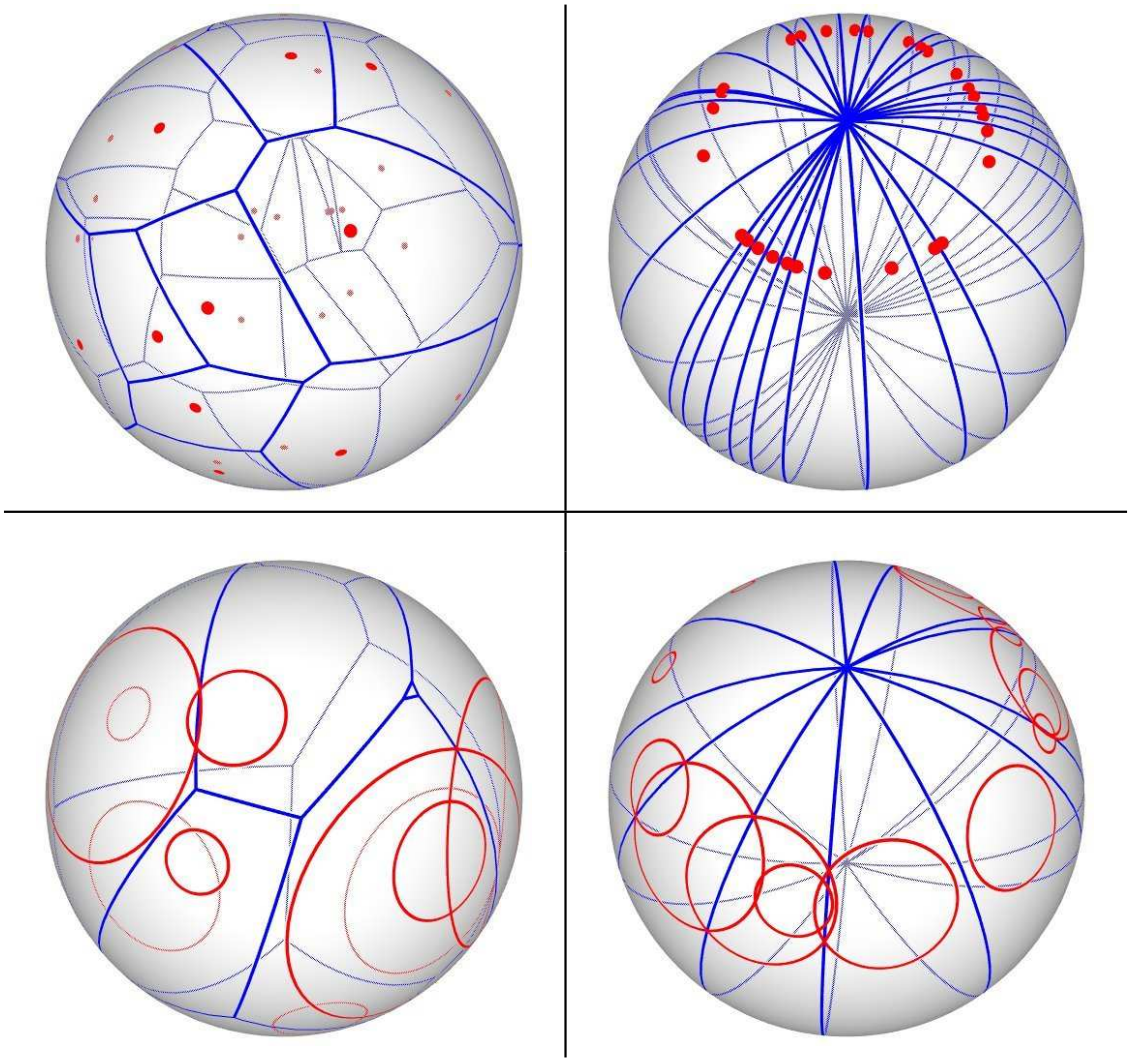
Examples of Available Diagrams

More Diagrams of Linear Objects



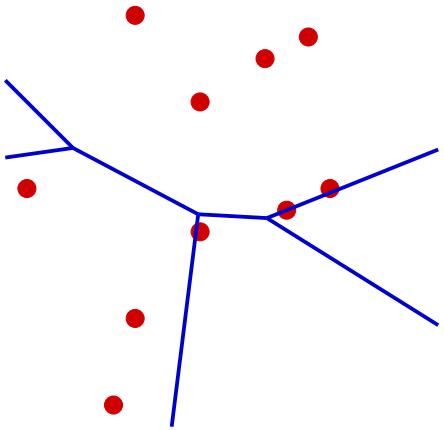
Examples of Available Diagrams

On the Sphere

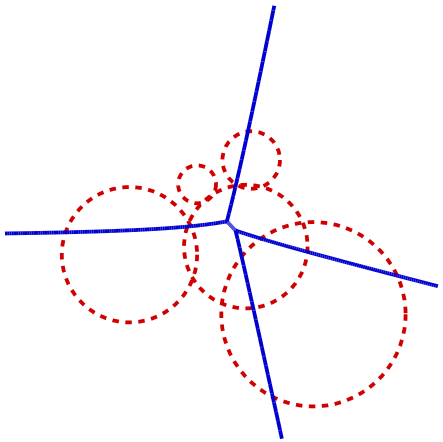


Examples of Available Diagrams

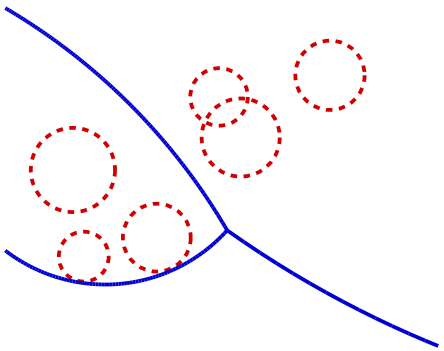
Farthest-site Voronoi Diagrams (by constructing upper envelopes)



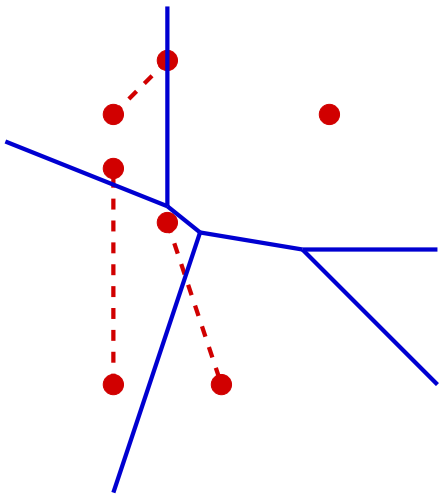
Farthest Voronoi diagram of points



Farthest Apollonius Voronoi diagram



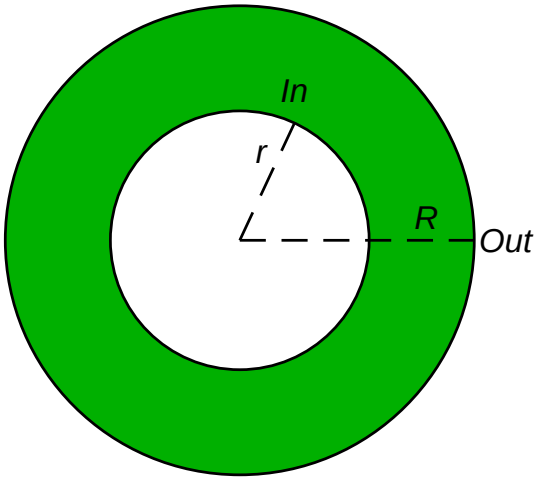
Farthest Möbius diagram

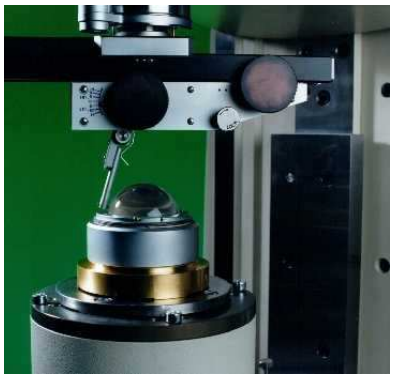


Farthest Voronoi diagram of segments

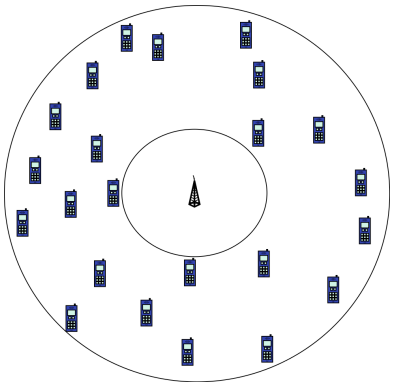
Application: Minimum-Width Annulus of Disks

- **Goal:** Given a set of disks in the plane, find an annulus of minimum width containing the disks
- Minimum-width annulus (MWA) has applications in tolerancing metrology and facility location
- We extended a known algorithm for computing a minimum-width annulus of points [Ebara et al., 1989] to disks





www.npl.co.uk/server.php



cgm.cs.mcgill.ca/~athens/cs507/
Projects/2004/Emory-Merryman

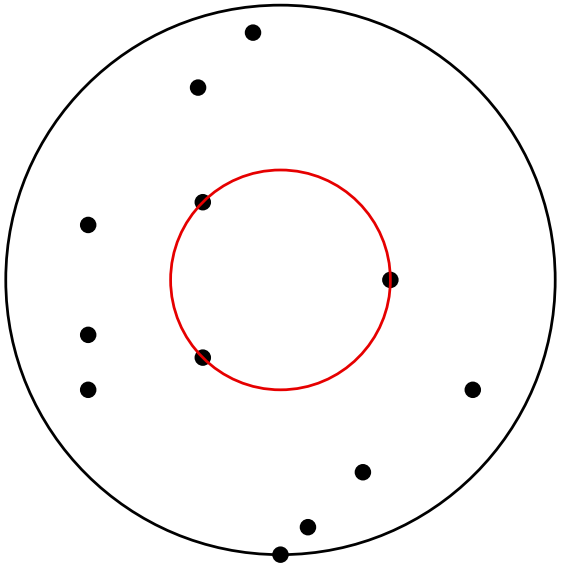
The Connection to Voronoi Diagrams

If MWA exists then it touches the objects in 4 points. There are 3 cases:

The Connection to Voronoi Diagrams

If MWA exists then it touches the objects in 4 points. There are 3 cases:

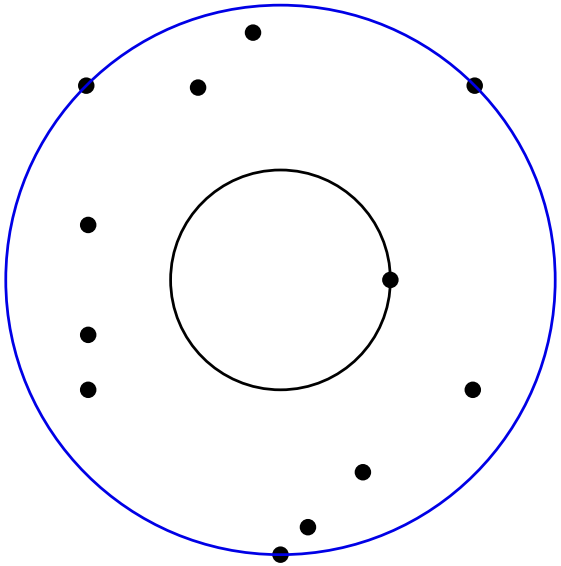
Inner circle touches 3 points — center is a nearest-site Voronoi vertex



The Connection to Voronoi Diagrams

If MWA exists then it touches the objects in 4 points. There are 3 cases:

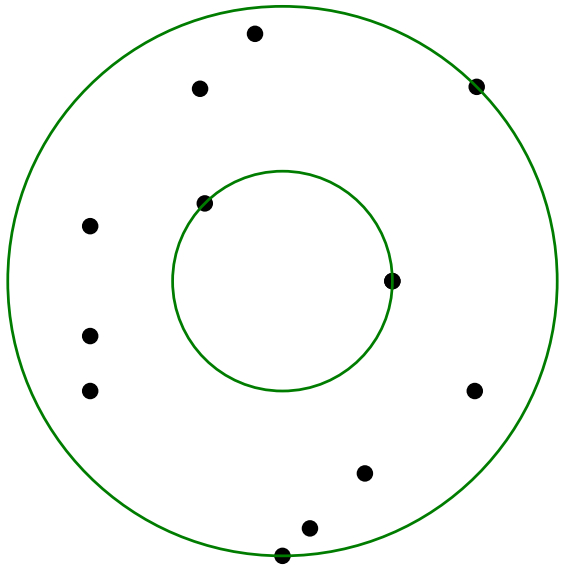
Outer circle touches 3 points — center is a farthest-site Voronoi vertex



The Connection to Voronoi Diagrams

If MWA exists then it touches the objects in 4 points. There are 3 cases:

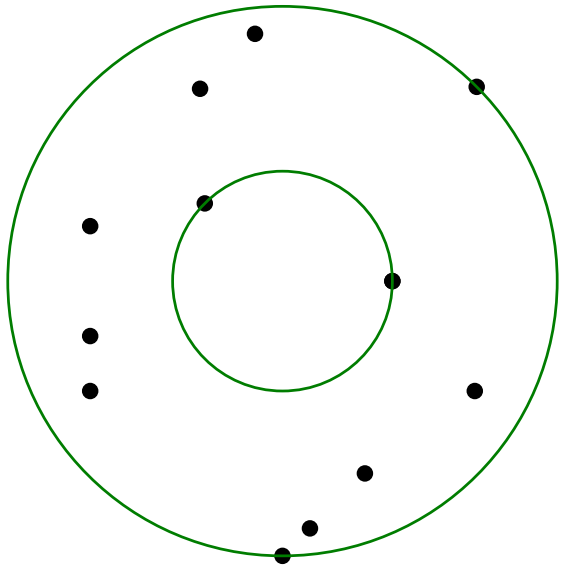
Both inner and outer circles touches $\geq$ 2 points — center is an intersection point between the diagrams (on edges of both diagrams)



The Connection to Voronoi Diagrams

If MWA exists then it touches the objects in 4 points. There are 3 cases:

Both inner and outer circles touches $\geq$ 2 points — center is an intersection point between the diagrams (on edges of both diagrams)

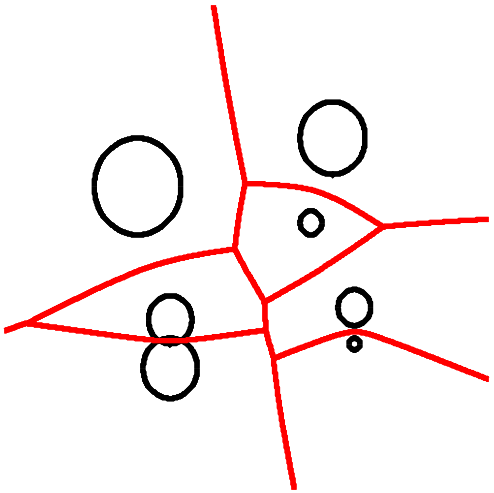


For points, only the third case occurs

The center of the MWA is a vertex of the overlay of the nearest-site and farthest-site Voronoi diagrams

MWA of Disks in the Plane

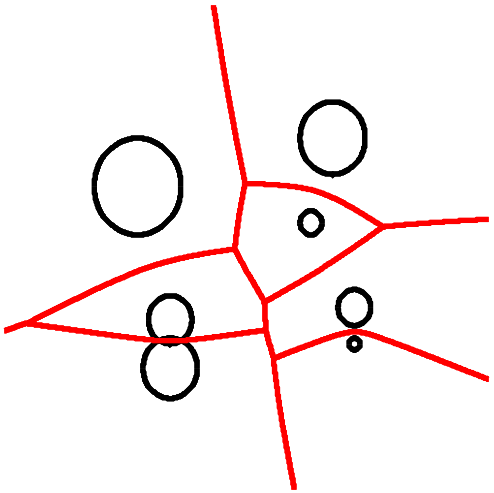
Nearest-site Voronoi is replaced by the Apollonius diagram



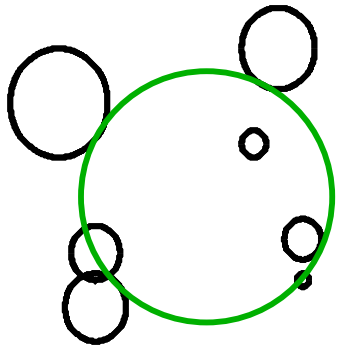
$$\delta(x, d_i) = ||x - c_i|| - r_i$$

MWA of Disks in the Plane

Nearest-site Voronoi is replaced by the Apollonius diagram



Farthest-site Apollonius diagram is not good in this case

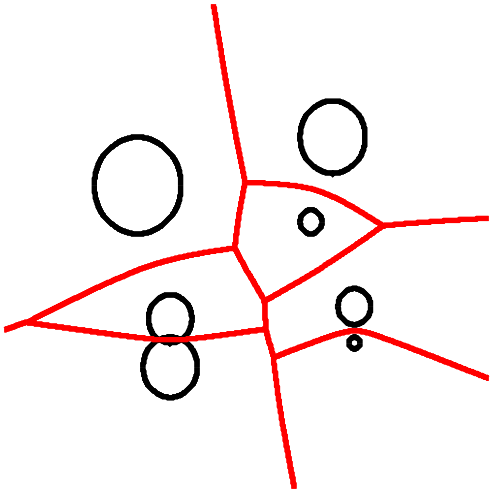


$$\delta(x, d_i) = ||x - c_i|| - r_i$$

We need to consider the farthest point of the disk from a point

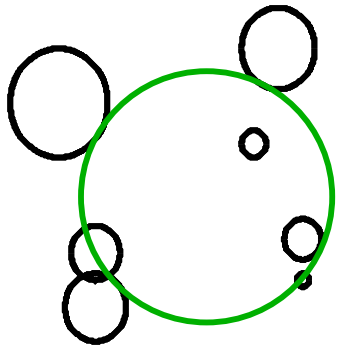
MWA of Disks in the Plane

Nearest-site Voronoi is replaced by the Apollonius diagram



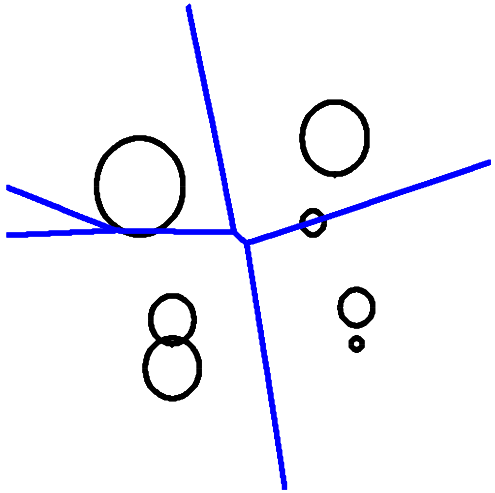
$\delta(x, d_i) = ||x - c_i|| - r_i$

Farthest-site Apollonius diagram is not good in this case



We need to consider the farthest point of the disk from a point

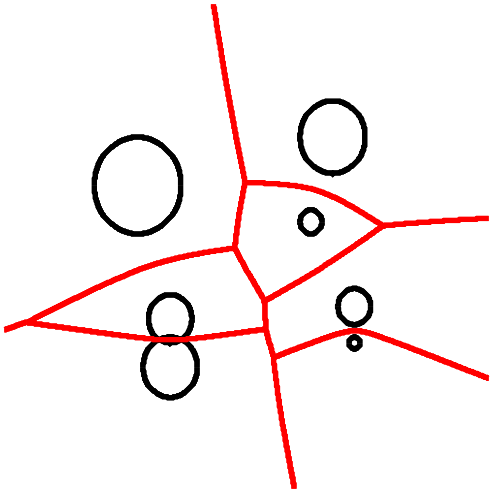
Farthest-Point Farthest-Site VD replaces the farthest-site VD



$\delta(x, d_i) = ||x - c_i|| + r_i$

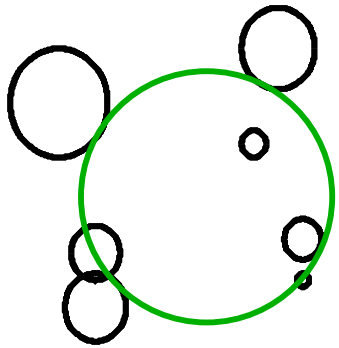
MWA of Disks in the Plane

Nearest-site Voronoi is replaced by the Apollonius diagram



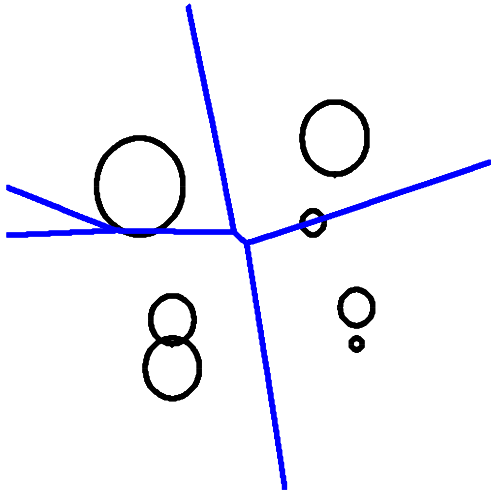
$$\delta(x, d_i) = ||x - c_i|| - r_i$$

Farthest-site Apollonius diagram is not good in this case



We need to consider the farthest point of the disk from a point

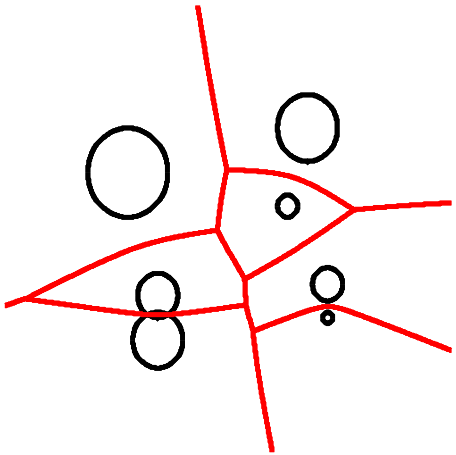
Farthest-Point Farthest-Site VD replaces the farthest-site VD

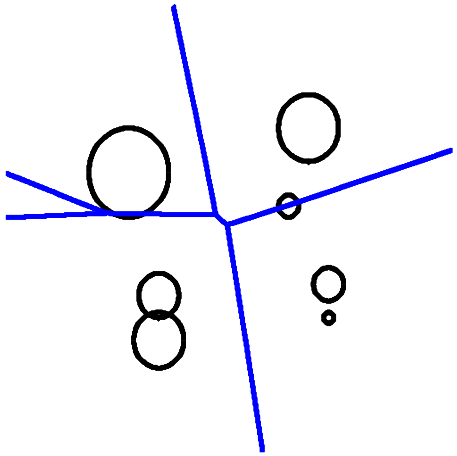


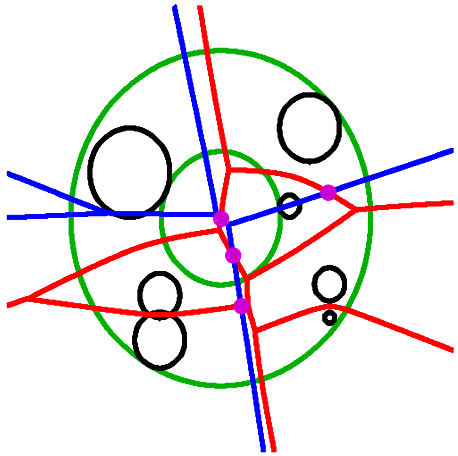
$$\delta(x, d_i) = ||x - c_i|| + r_i$$

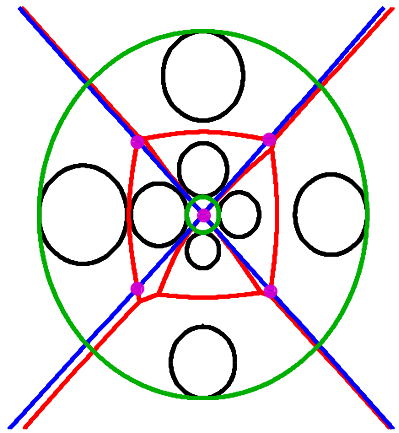
Farthest-point farthest-site is a farthest-site Apollonius with negative radii and was easily produced using our framework

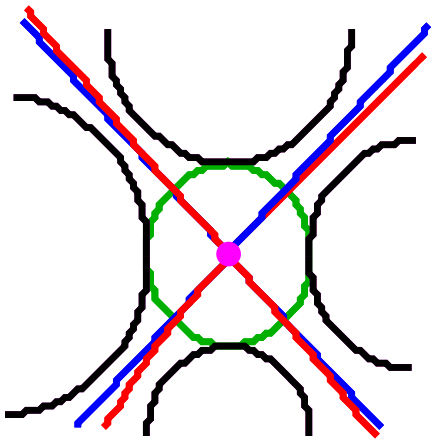
MWA of Disks in the Plane, running times











No. Disks	Time (secs)	V	E	F
50	10.741	126	213	88
100	26.994	238	395	158
200	62.968	416	659	244
500	185.244	775	1174	400
1000	405.405	1242	1894	653

Voronoi diagram

max planck institut informatik

Thank you Ophir, for providing the slides.