CGAL 3D Triangulations in Periodic Spaces

Manuel Caroli, Nico Kruithof, Monique Teillaud

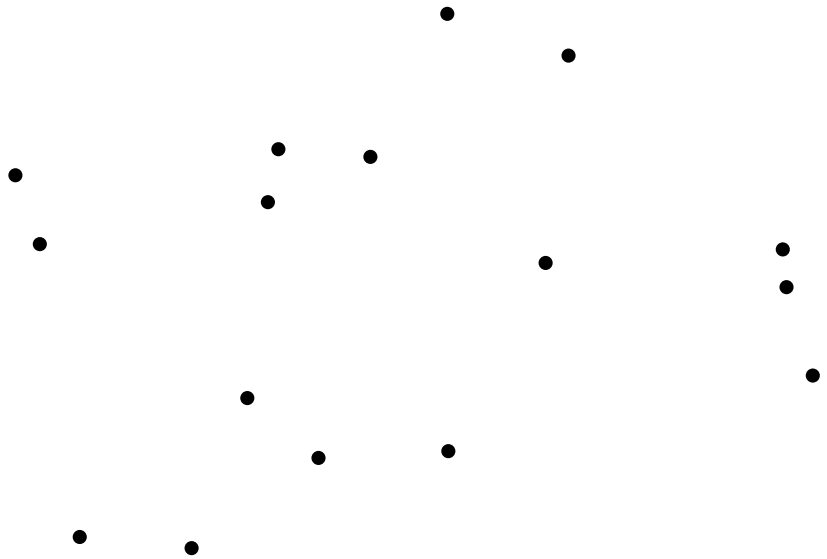
Workshop on Geometric Computing in Periodic Spaces

20th October 2008

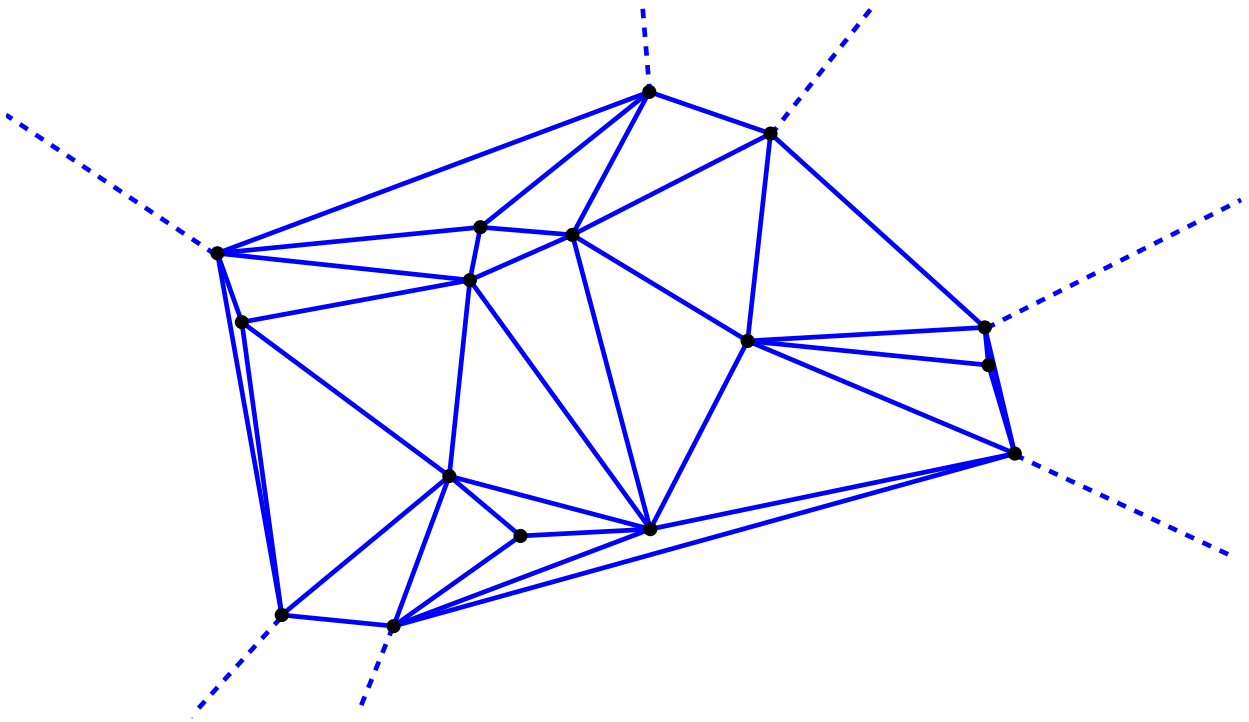
Outline

Outline

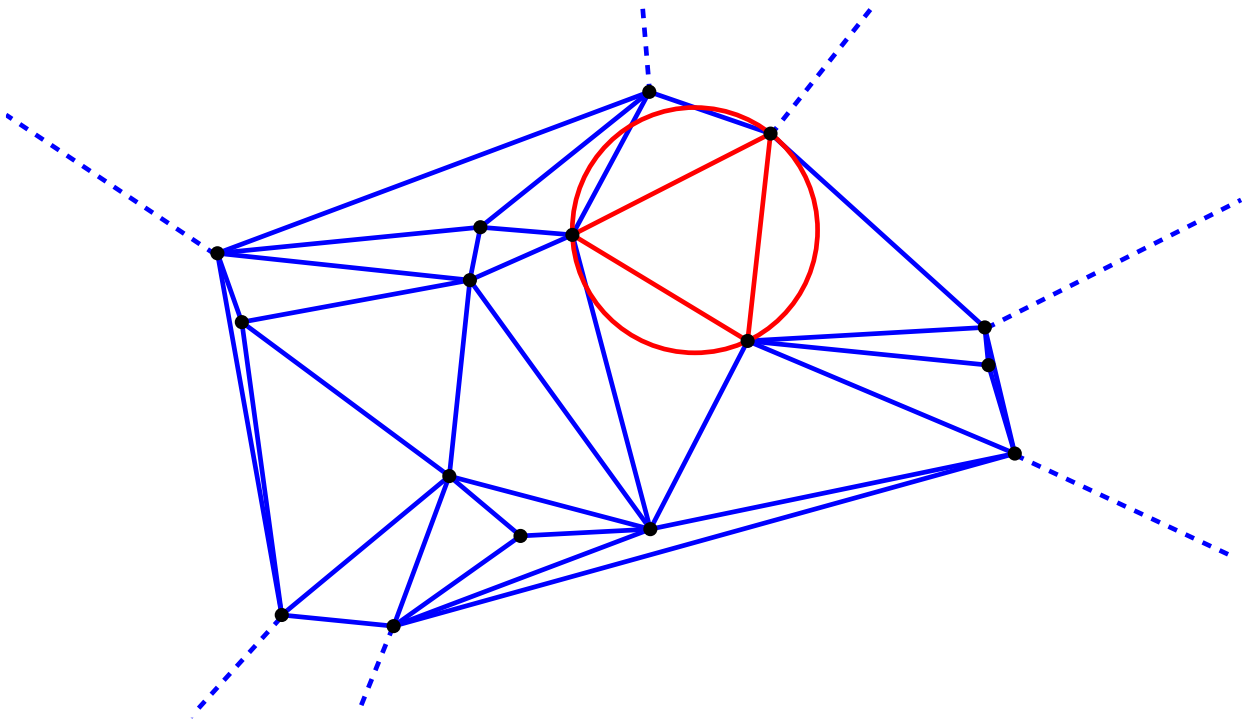
Delaunay triangulation



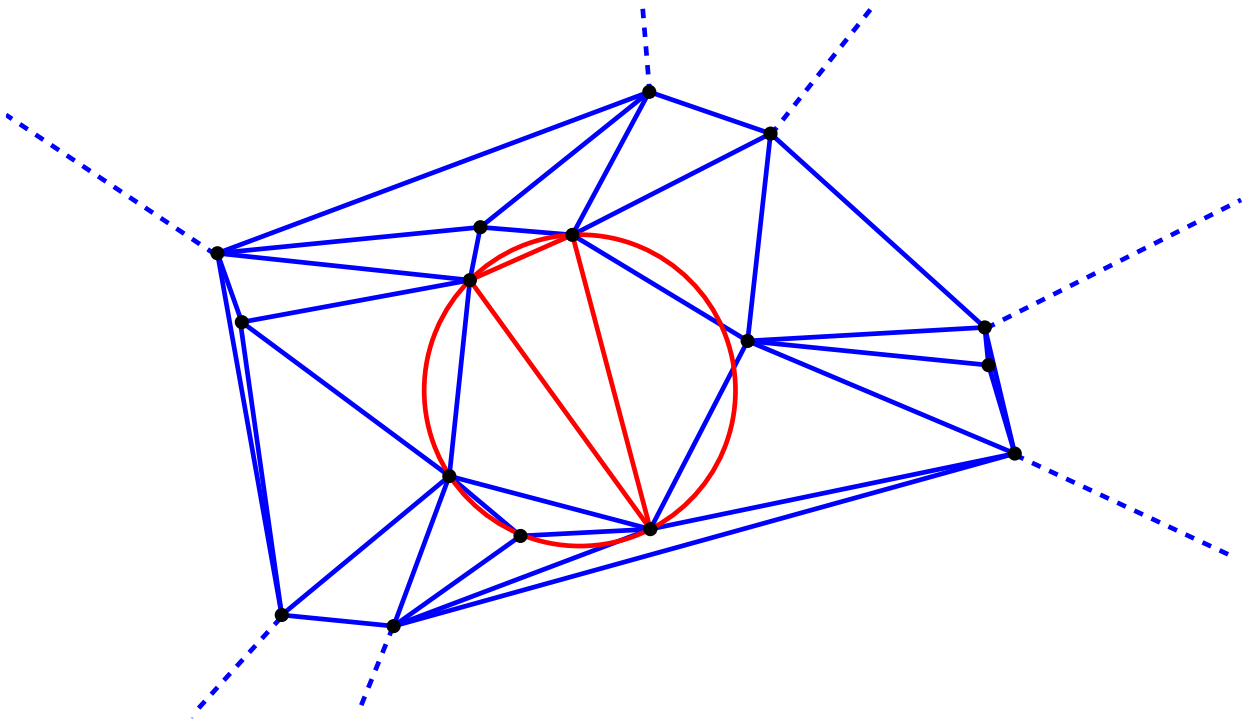
Delaunay triangulation



Delaunay triangulation

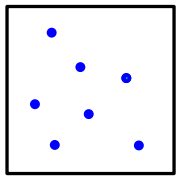


Delaunay triangulation



Periodic space $\mathbb{T}^2$

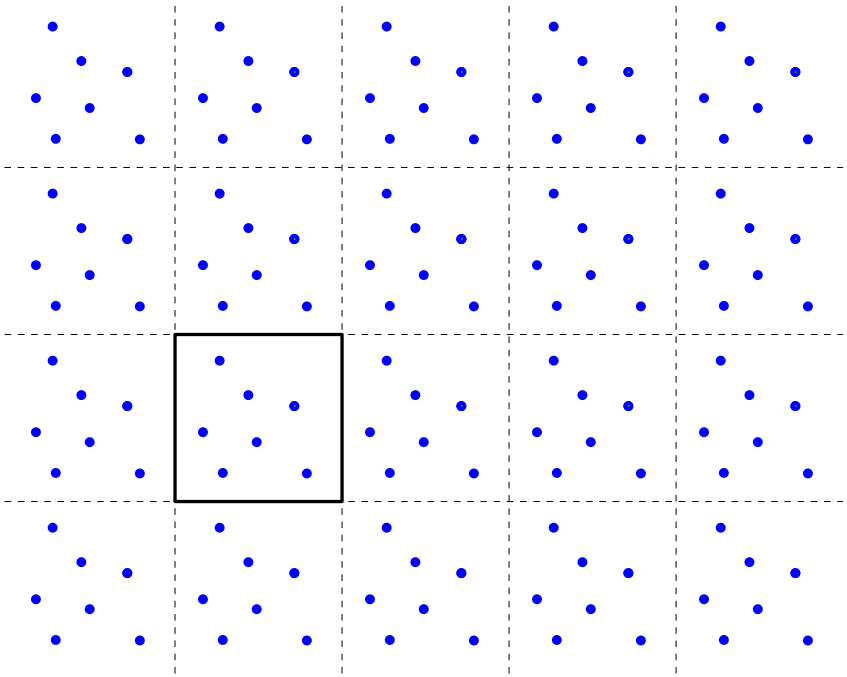
- $\mathbb{T}^2 = \mathbb{R}^2/\mathbb{Z}^2$, $\pi := \mathbb{R}^2 \rightarrow \mathbb{T}^2$



$\mathcal{S}$

Periodic space $\mathbb{T}^2$

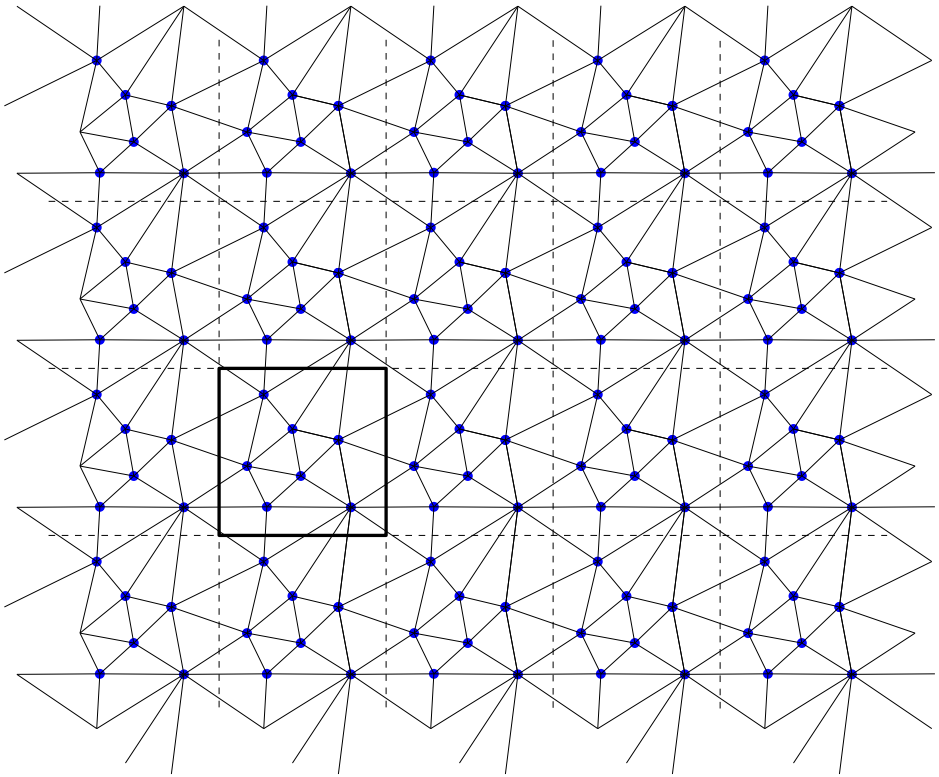
- $\mathbb{T}^2 = \mathbb{R}^2/\mathbb{Z}^2$, $\pi := \mathbb{R}^2 \to \mathbb{T}^2$
- $\pi^{-1}(p)$ maps p onto a regular point lattice



$\pi^{-1}(\mathcal{S})$

Periodic space $\mathbb{T}^2$

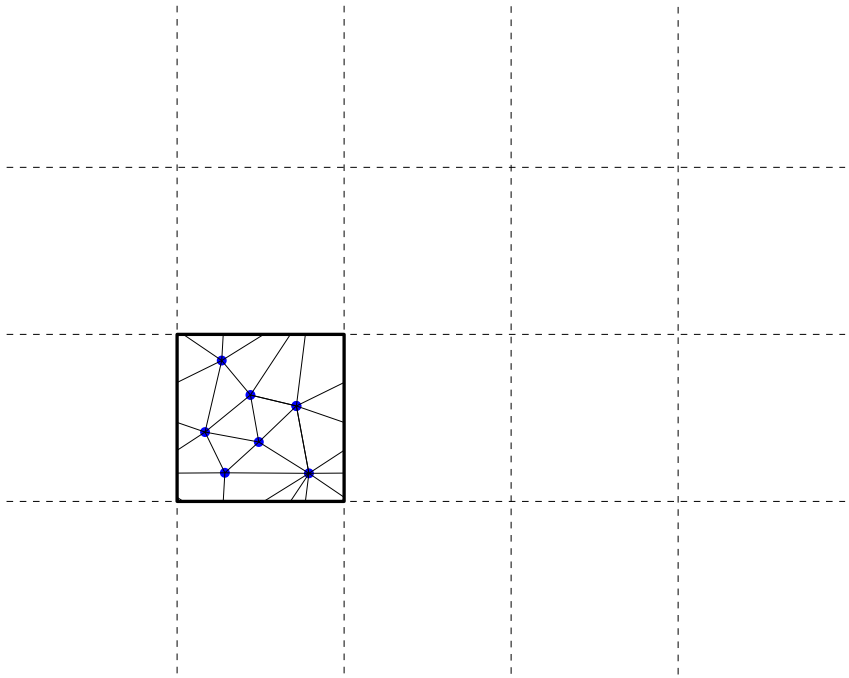
- $\mathbb{T}^2 = \mathbb{R}^2/\mathbb{Z}^2$, $\pi := \mathbb{R}^2 \rightarrow \mathbb{T}^2$
- $\pi^{-1}(p)$ maps p onto a regular point lattice



$DT(\pi^{-1}(\mathcal{S}))$

Periodic space $\mathbb{T}^2$

- $\mathbb{T}^2 = \mathbb{R}^2/\mathbb{Z}^2$, $\pi := \mathbb{R}^2 \rightarrow \mathbb{T}^2$
- $\pi^{-1}(p)$ maps p onto a regular point lattice



$$\pi(DT(\pi^{-1}(\mathcal{S})))$$

Triangulation

Definition

A *Triangulation of a point set* $\mathcal{S}$ is
a simplicial complex K in
a topological space $\mathbb{X}$, if

1. each point in $\mathcal{S}$ is a vertex of K
2. $\bigcup_{\sigma \in K} \sigma$ is homeomorphic to $\mathbb{X}$.

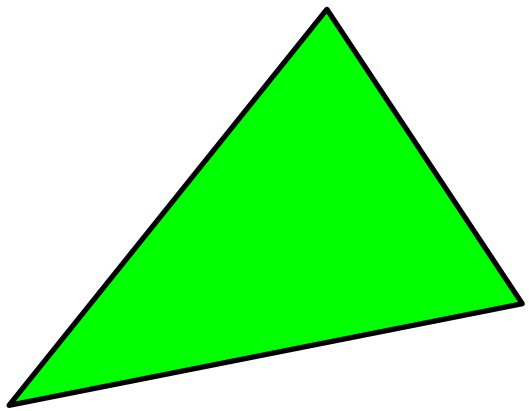
Simplicial complex

Definition

A *simplicial complex* K is

a finite set of simplices such that

1. $\sigma \in K, \tau \leq \sigma \Rightarrow \tau \in K$
2. $\sigma, \sigma' \in K \Rightarrow \sigma \cap \sigma' \leq \sigma, \sigma'$



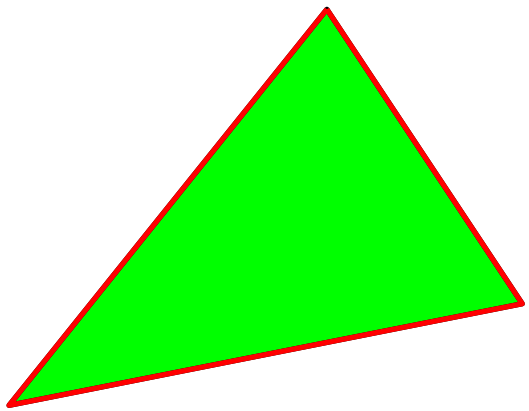
Simplicial complex

Definition

A *simplicial complex* K is

a finite set of simplices such that

1. $\sigma \in K, \tau \leq \sigma \Rightarrow \tau \in K$
2. $\sigma, \sigma' \in K \Rightarrow \sigma \cap \sigma' \leq \sigma, \sigma'$



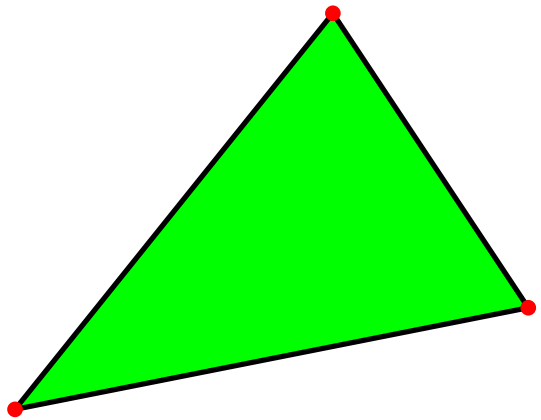
Simplicial complex

Definition

A *simplicial complex* K is

a finite set of simplices such that

1. $\sigma \in K, \tau \leq \sigma \Rightarrow \tau \in K$
2. $\sigma, \sigma' \in K \Rightarrow \sigma \cap \sigma' \leq \sigma, \sigma'$



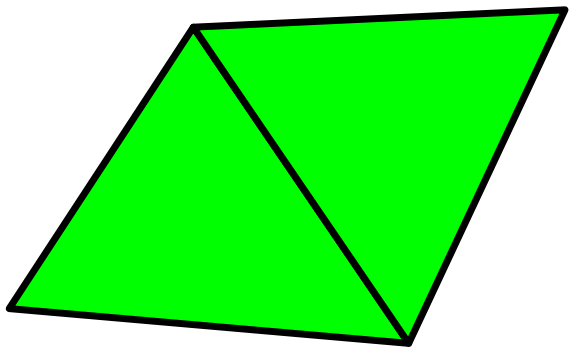
Simplicial complex

Definition

A *simplicial complex* K is

a finite set of simplices such that

1. $\sigma \in K, \tau \leq \sigma \Rightarrow \tau \in K$
2. $\sigma, \sigma' \in K \Rightarrow \sigma \cap \sigma' \leq \sigma, \sigma'$



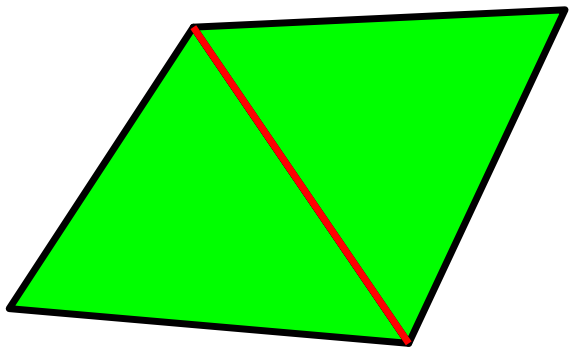
Simplicial complex

Definition

A *simplicial complex* K is

a finite set of simplices such that

1. $\sigma \in K, \tau \leq \sigma \Rightarrow \tau \in K$
2. $\sigma, \sigma' \in K \Rightarrow \sigma \cap \sigma' \leq \sigma, \sigma'$

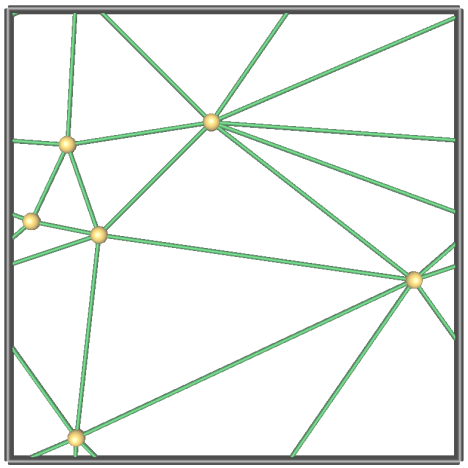


Delaunay triangulation in $\mathbb{T}^3$.

Definition

We call $DT_{\mathbb{T}}(\mathcal{S}) := \pi(DT(\pi^{-1}(\mathcal{S})))$ a *Delaunay tessellation*.

We call $DT_{\mathbb{T}}(\mathcal{S})$ a Delaunay *triangulation* if it is a simplicial complex.

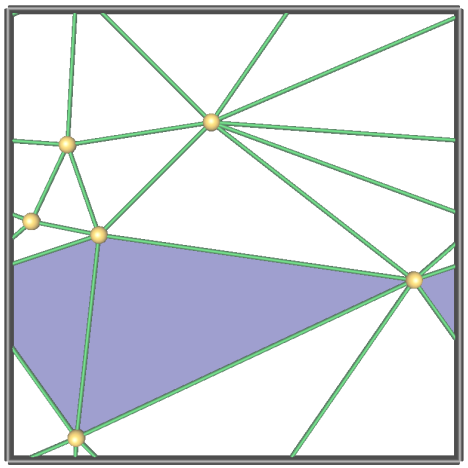


Delaunay triangulation in $\mathbb{T}^3$.

Definition

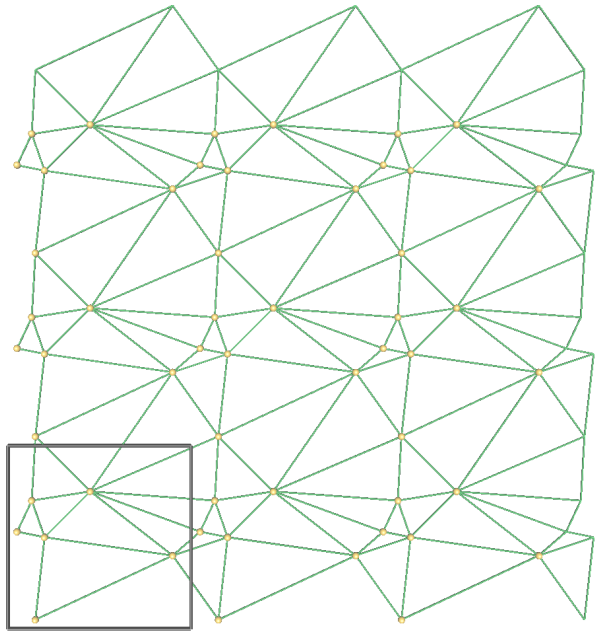
We call $DT_{\mathbb{T}}(\mathcal{S}) := \pi(DT(\pi^{-1}(\mathcal{S})))$ a *Delaunay tessellation*.

We call $DT_{\mathbb{T}}(\mathcal{S})$ a Delaunay *triangulation* if it is a simplicial complex.

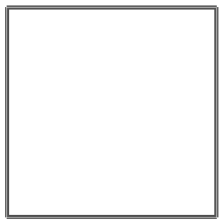


Covering spaces

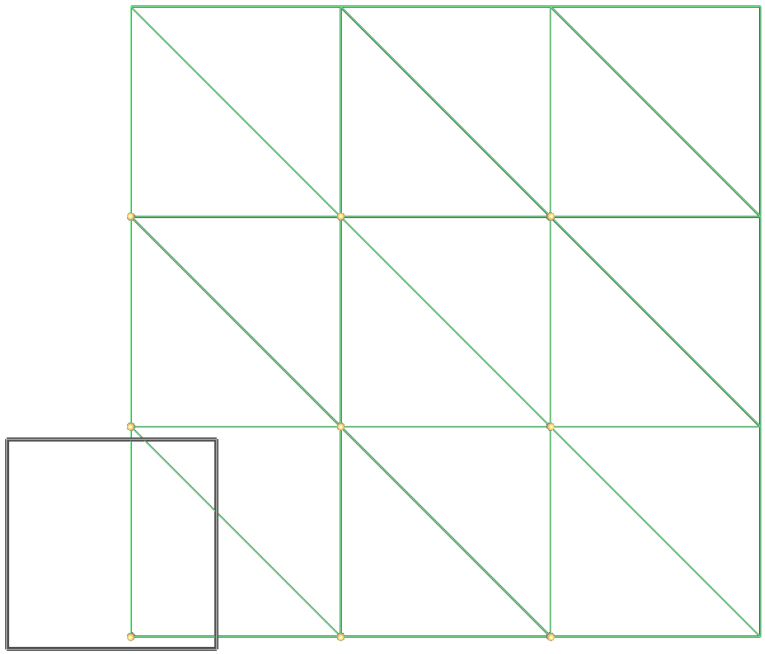
Solution: Computing in $\mathbb{T}^2_3 := \mathbb{R}^2/3\mathbb{Z}^3$
with point set $\pi_3(\pi^{-1}(\mathcal{S}))$



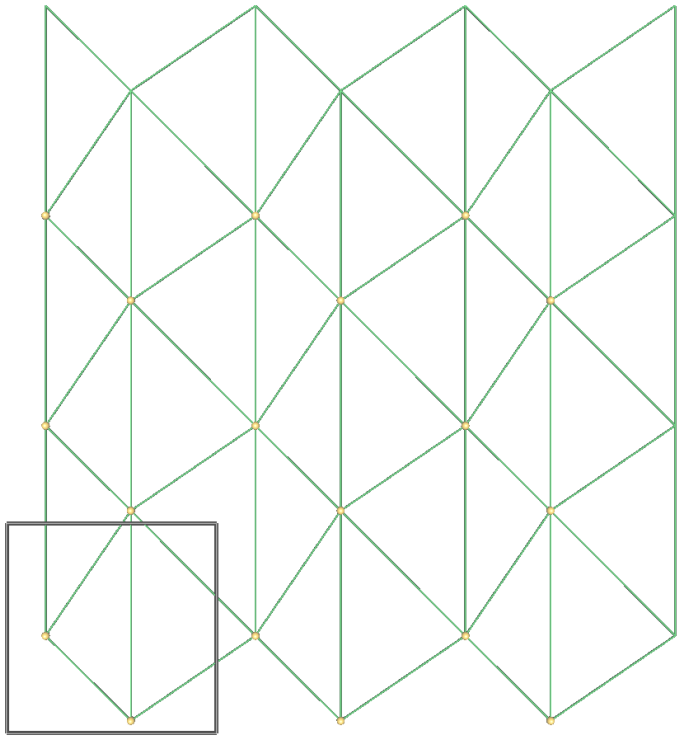
Building up a triangulation



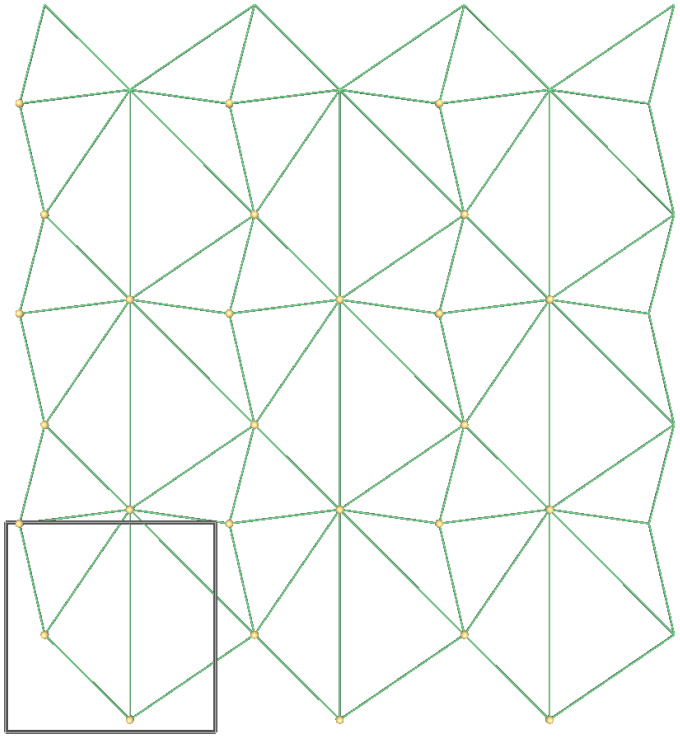
Building up a triangulation



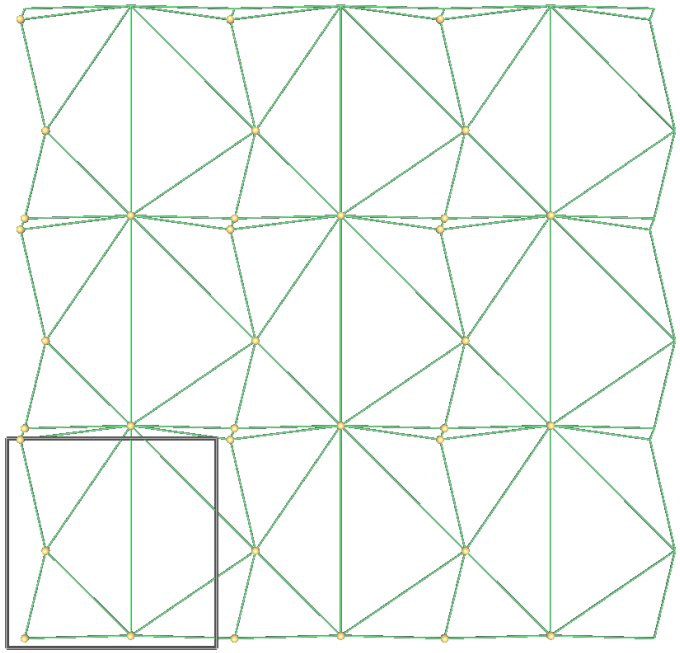
Building up a triangulation



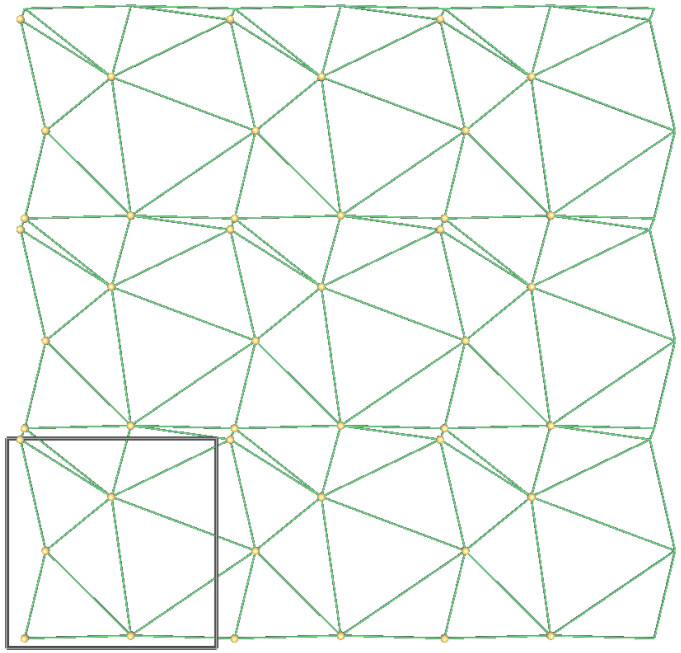
Building up a triangulation



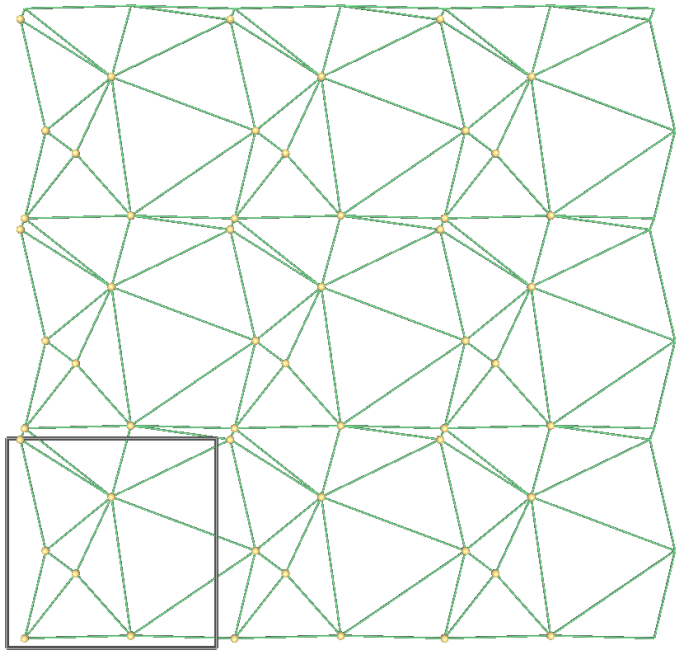
Building up a triangulation



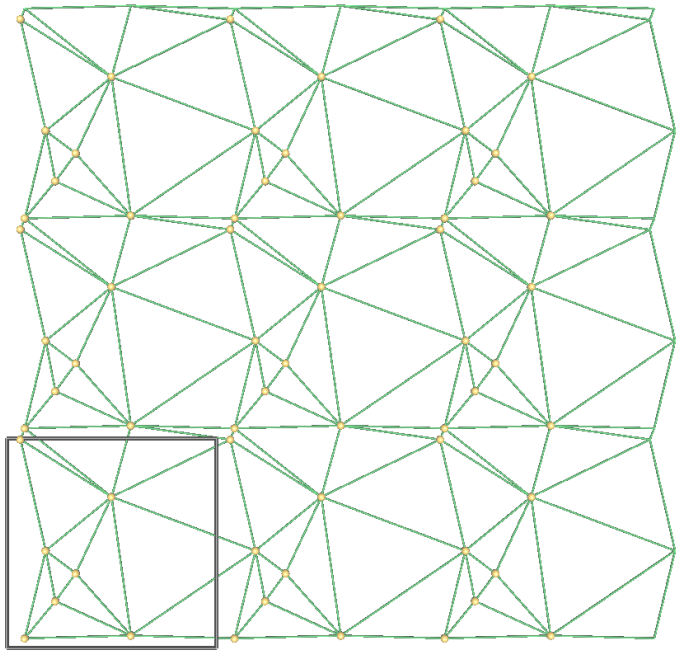
Building up a triangulation



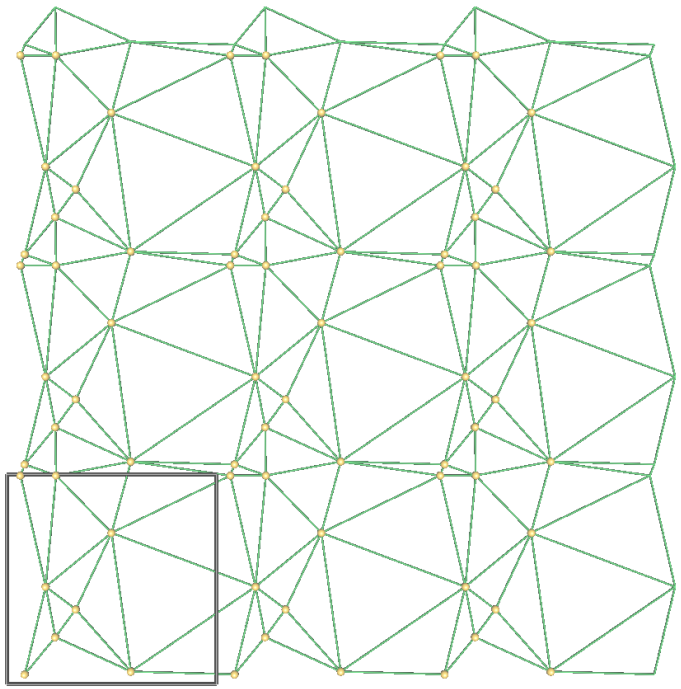
Building up a triangulation



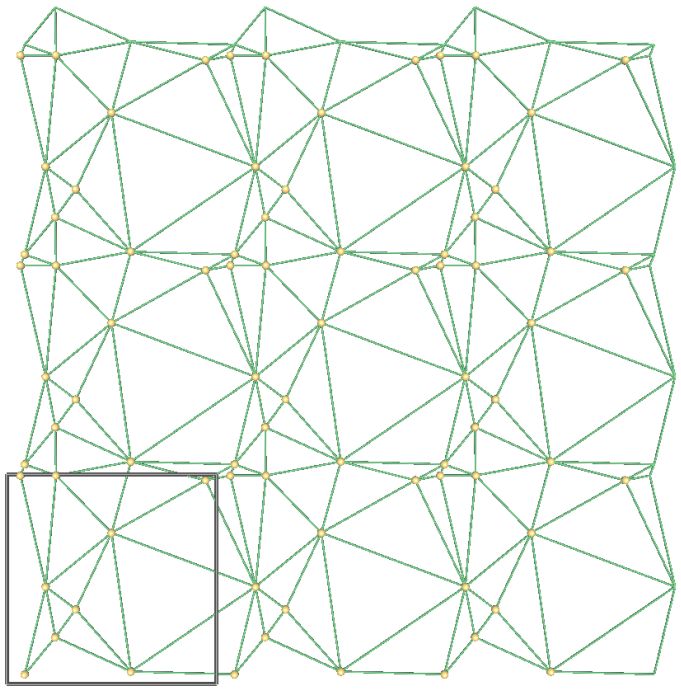
Building up a triangulation



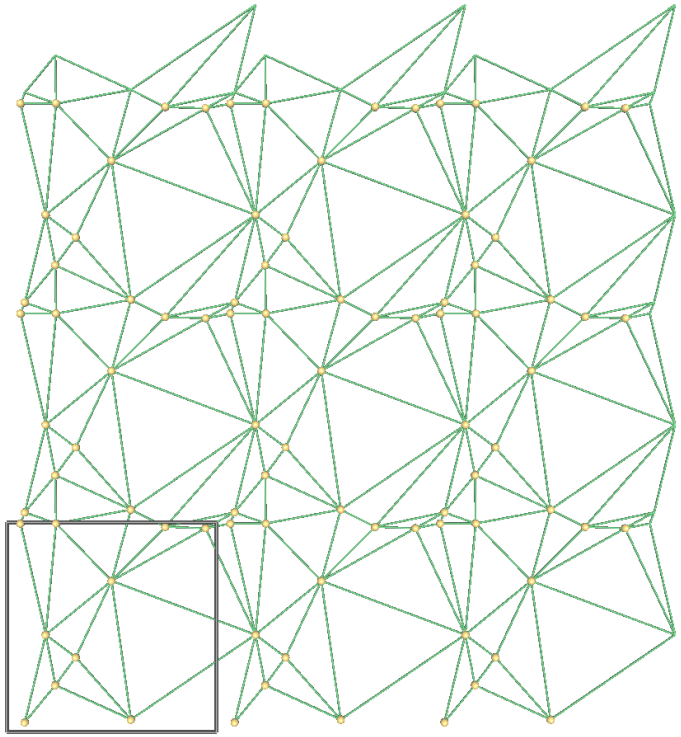
Building up a triangulation



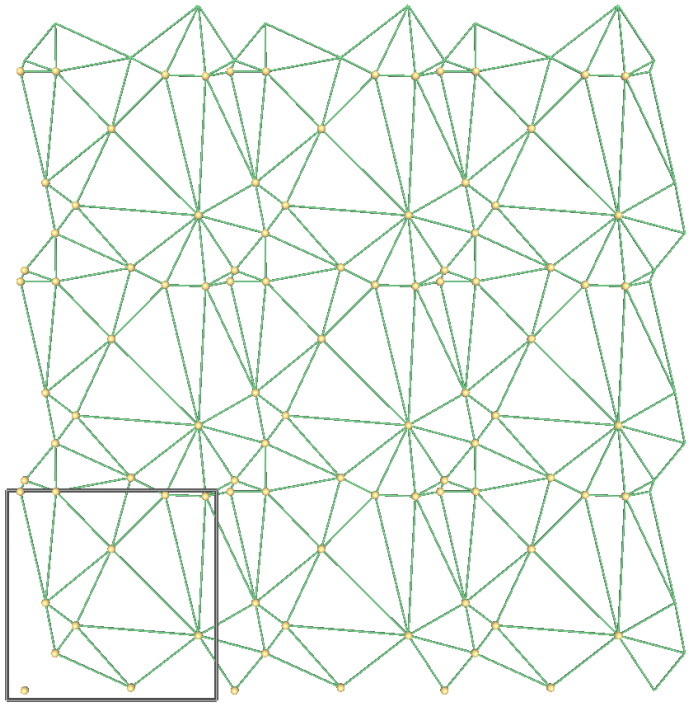
Building up a triangulation



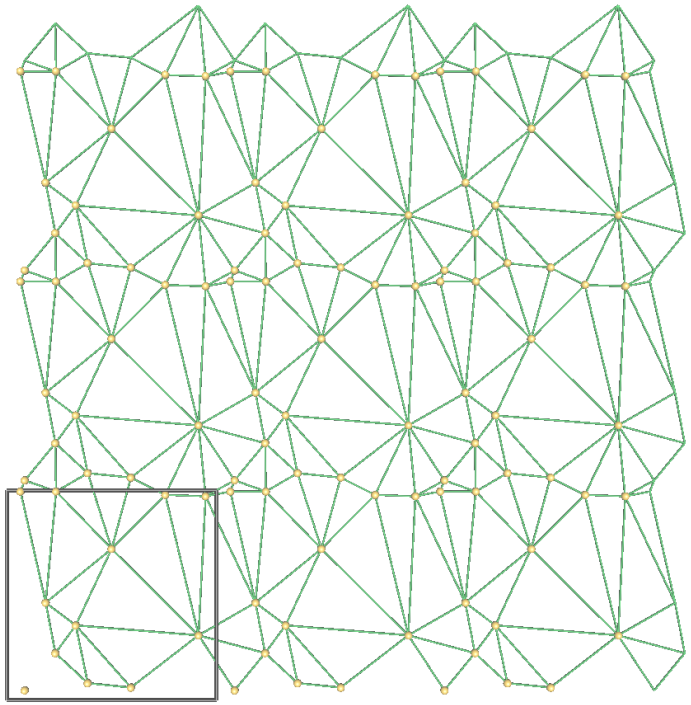
Building up a triangulation



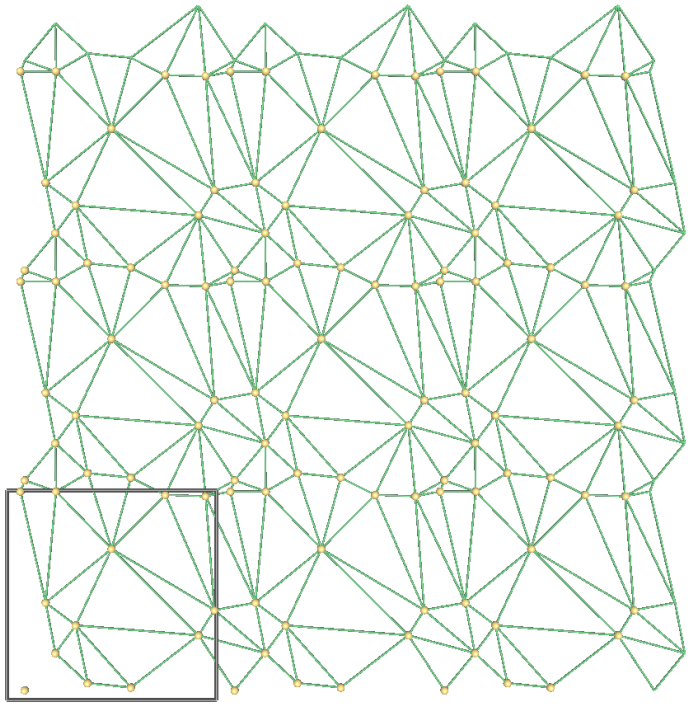
Building up a triangulation



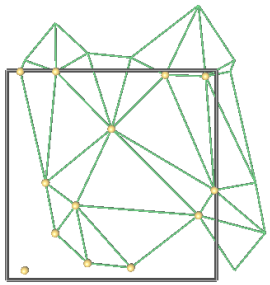
Building up a triangulation



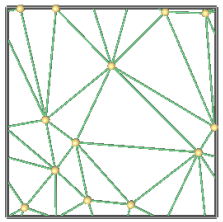
Building up a triangulation



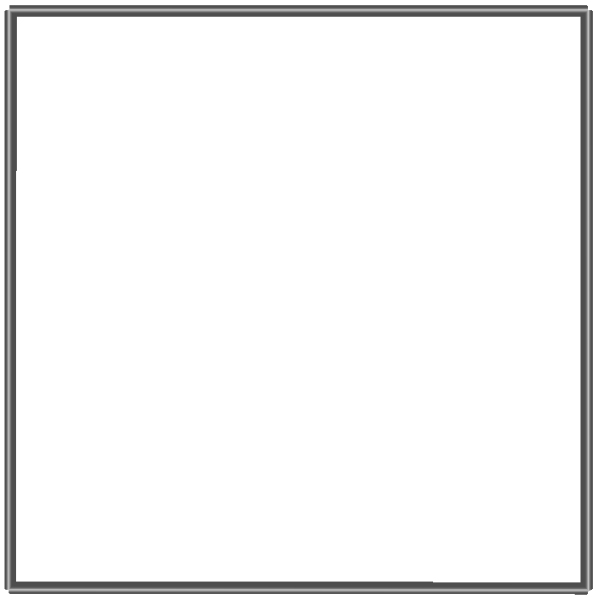
Building up a triangulation



Building up a triangulation

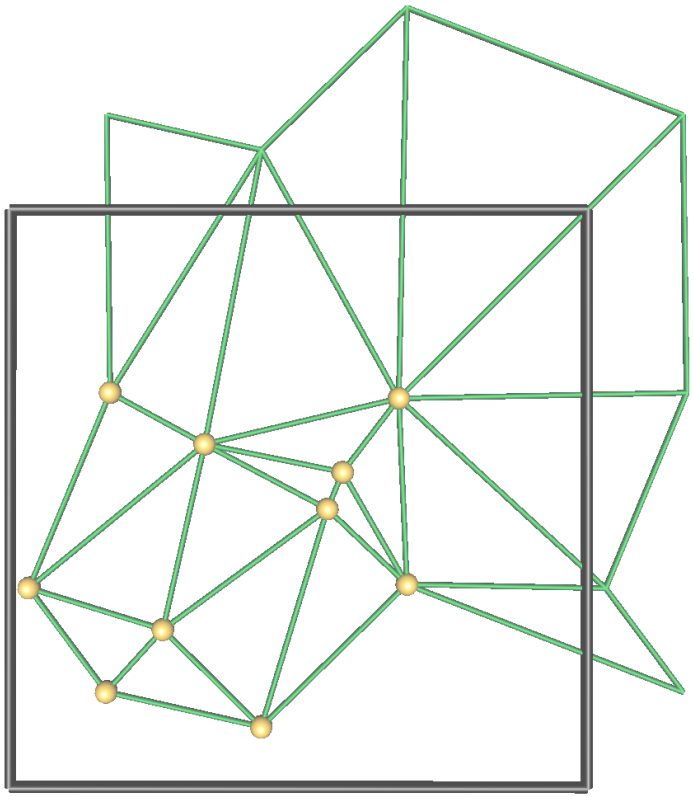


Periodic space $\mathbb{T}^2$



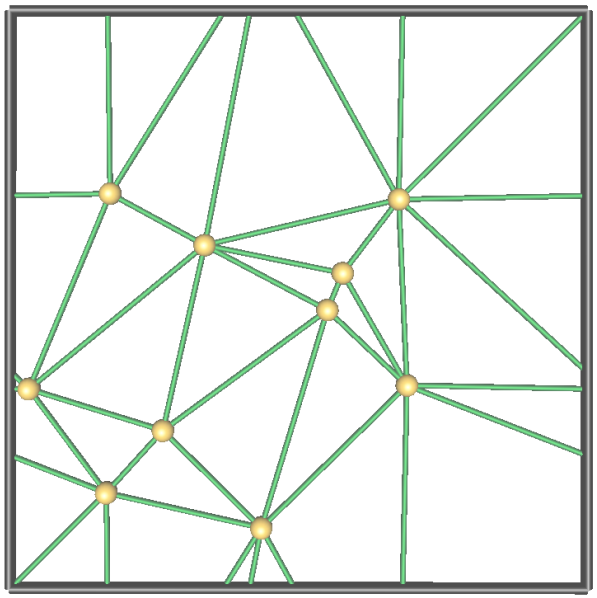
$$\mathbb{T}^2 = \mathbb{R}^2/\mathbb{Z}^2$$

Periodic space $\mathbb{T}^2$



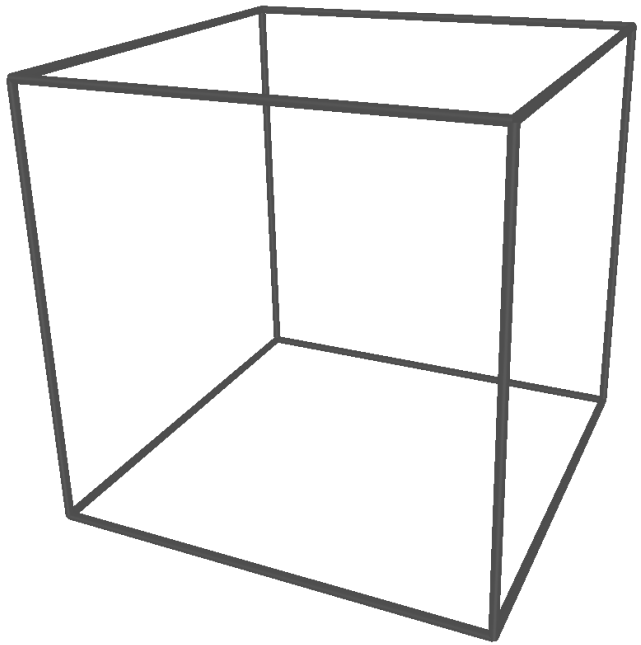
$$\mathbb{T}^2 = \mathbb{R}^2/\mathbb{Z}^2$$

Periodic space $\mathbb{T}^2$



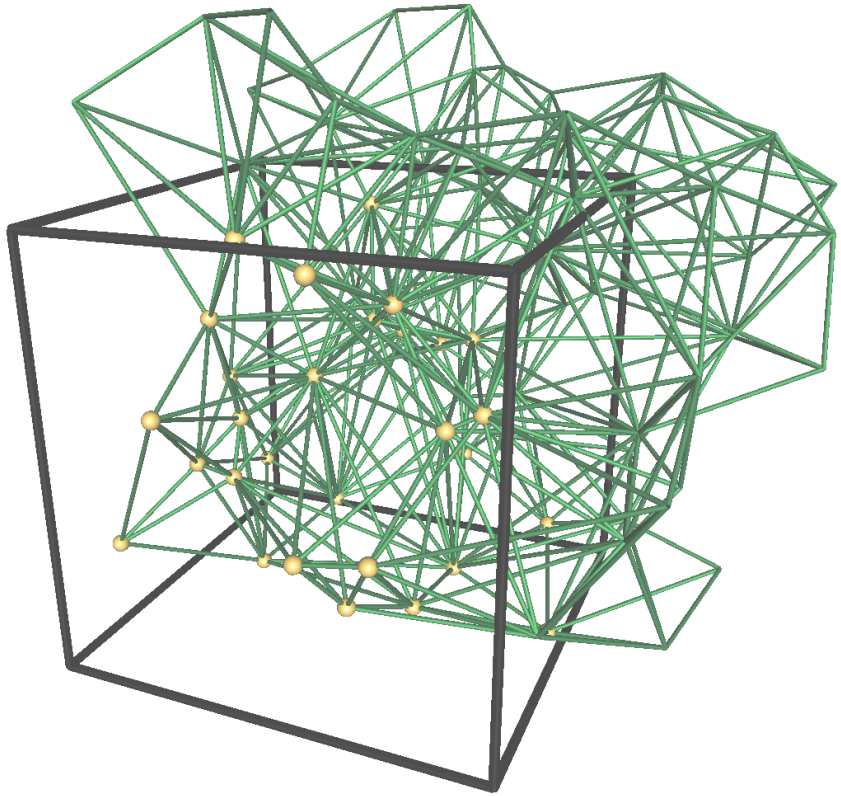
$$\mathbb{T}^2 = \mathbb{R}^2/\mathbb{Z}^2$$

Periodic space $\mathbb{T}^3$



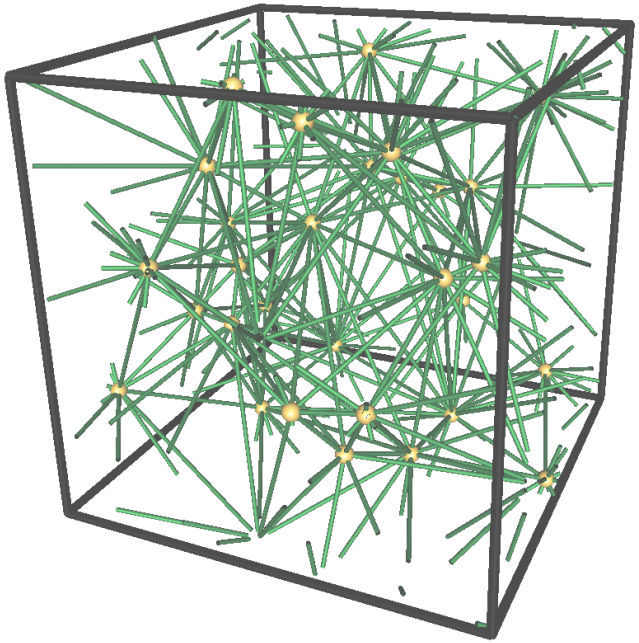
$$\mathbb{T}^3 = \mathbb{R}^3/\mathbb{Z}^3$$

Periodic space $\mathbb{T}^3$



$$\mathbb{T}^3 = \mathbb{R}^3/\mathbb{Z}^3$$

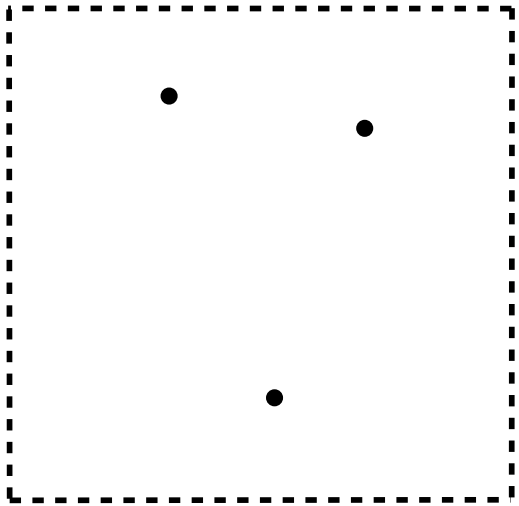
Periodic space $\mathbb{T}^3$



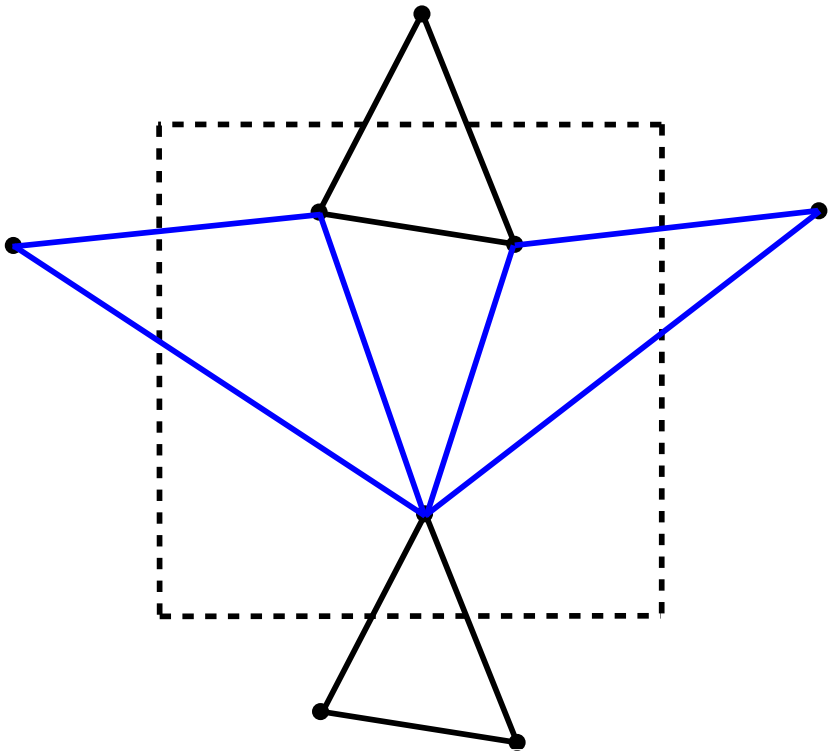
$$\mathbb{T}^3 = \mathbb{R}^3/\mathbb{Z}^3$$

Outline

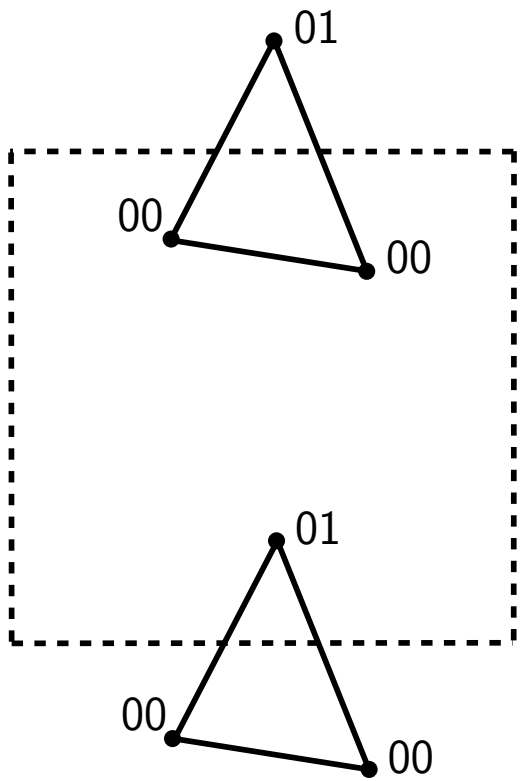
Offsets



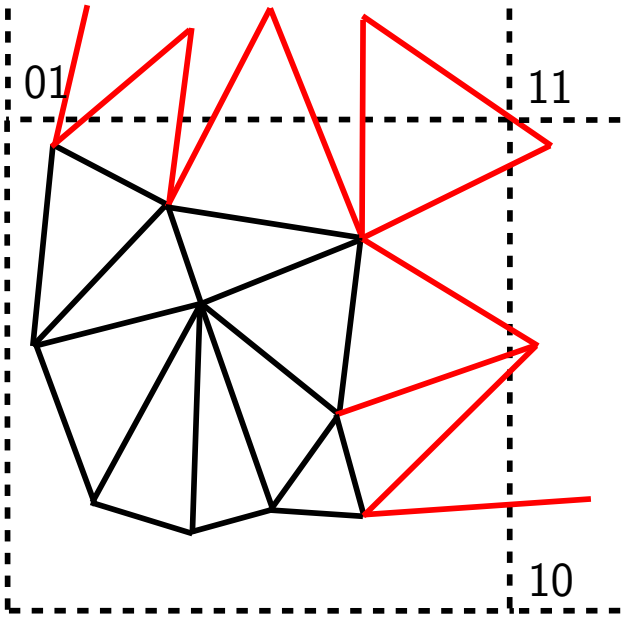
Offsets



Offsets



Offsets



further differences to $\mathbb{R}^3$.

- Offsets:
  - `Periodic_3_offset_3` ($\mathbb{Z}^3$)
  - `Cell_handle ch`:
    `ch->vertex(i)`, $0 \leq i < 4$
    `ch->offset(i)`, $0 \leq i < 4$

further differences to $\mathbb{R}^3$.

- Offsets:
  - `Periodic_3_offset_3` ($\mathbb{Z}^3$)
  - `Cell_handle ch`:
    `ch->vertex(i)`, $0 \leq i < 4$
    `ch->offset(i)`, $0 \leq i < 4$
- no infinite vertex:
  - `is_infinite()`
  - `Finite_*_iterator`
  - ...

further differences to $\mathbb{R}^3$.

- Offsets:
  - `Periodic_3_offset_3` ($\mathbb{Z}^3$)
  - `Cell_handle ch`:
    `ch->vertex(i)`, $0 \leq i < 4$
    `ch->offset(i)`, $0 \leq i < 4$
- no infinite vertex:
  - `is_infinite()`
  - `Finite_*_iterator`
  - ...
- no degenerate dimensions:
  - `dimension()`
  - no side of circle test

Triangulation traits

`Triangulation_3` parameterized by `Geometric_traits`:

- Geometric primitives: points, segments, triangles ...
- Number types, basic operations
- Geometric predicates: side of sphere, ...
- Geometric constructors: construct triangle, ...

```
Triangulation_3<
    TriangulationTraits_,
    TriangulationDataStructure_3 >
```

Triangulation traits

`Triangulation_3` parameterized by `Geometric_traits`:

- Geometric primitives: points, segments, triangles ...
- Number types, basic operations
- Geometric predicates: side of sphere, ...
- Geometric constructors: construct triangle, ...

```
Periodic_3_Triangulation_3<
    Periodic_3TriangulationTraits_3,
    TriangulationDataStructure_3 >
```

Periodic traits

- side_of_sphere, orientation, etc. with offsets
- Delaunay triangulation traits:
  - orientation
  - coplanar orientation
  - side of sphere
  - side of circle
  - ...

Periodic traits

- side_of_sphere, orientation, etc. with offsets
- Periodic Delaunay triangulation traits:
  - orientation
  - ~~coplanar orientation~~
  - side of sphere
  - ~~side of circle~~
  - ...

Periodic traits

- side_of_sphere, orientation, etc. with offsets
- Periodic Delaunay triangulation traits:
  - orientation
  - ~~coplanar orientation~~
  - side of sphere
  - ~~side of circle~~
  - ...
- Offsets:

$$\texttt{pred}(p_0 , p_1 , \ldots, p_k)$$

Periodic traits

- side_of_sphere, orientation, etc. with offsets
- Periodic Delaunay triangulation traits:
  - orientation
  - ~~coplanar orientation~~
  - side of sphere
  - ~~side of circle~~
  - ...
- Offsets:

  `periodic_pred`$(p_0, p_1, \ldots, p_k, o_0, o_1, \ldots, o_k)$
  $:=$`pred`$(p_0 + o_0, p_1 + o_1, \ldots, p_k + o_k)$

Access

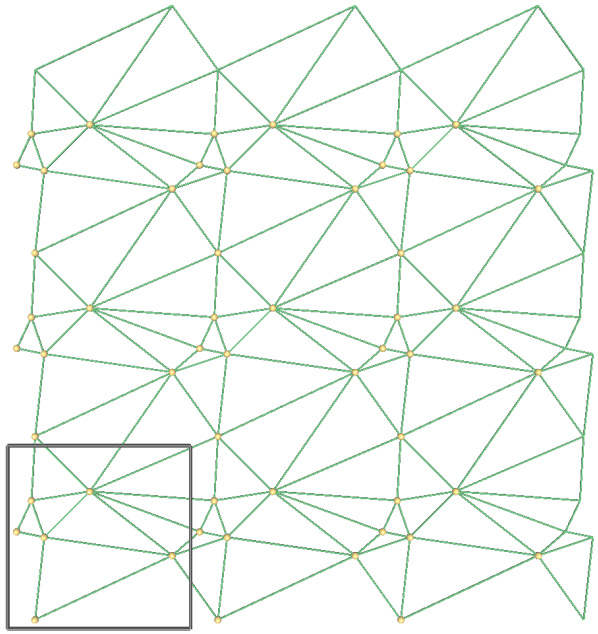
- Combinatorial iterators:
  - `Vertex_iterator`
  - `Edge_iterator`
  - `Facet_iterator`
  - `Cell_iterator`
- Return all primitives from the triangulation.
- Expectation: no periodic copies
- Inconsistencies if not simplicial complex

Access

- Combinatorial iterators:
  - `[...]_iterator`
- Return all primitives from the triangulation.
- Expectation: no periodic copies
- Inconsistencies if not simplicial complex:

Access

- Combinatorial iterators:
  - `[...]_iterator`
- Return all primitives from the triangulation.
- Expectation: no periodic copies
- Inconsistencies if not simplicial complex:



Access II

- Geometric iterators:
  - `Point_iterator`
  - `Segment_iterator`
  - `Triangle_iterator`
  - `Tetrahedron_iterator`

Access II

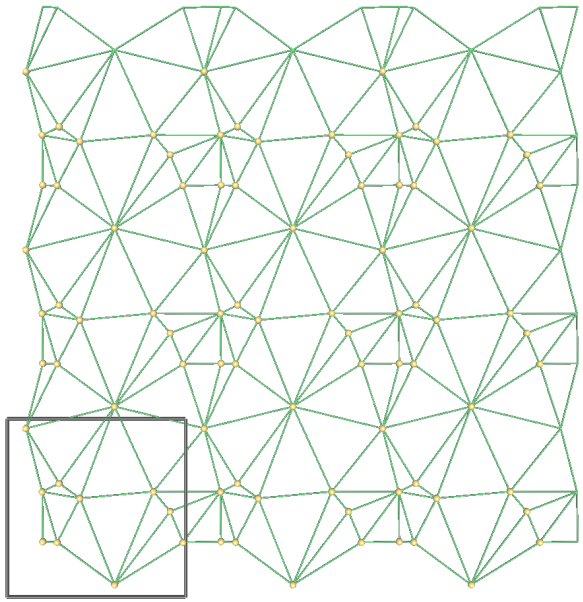
- Geometric iterators

Access II

- Geometric iterators
- Parameter `Iterator_type it`

Access II

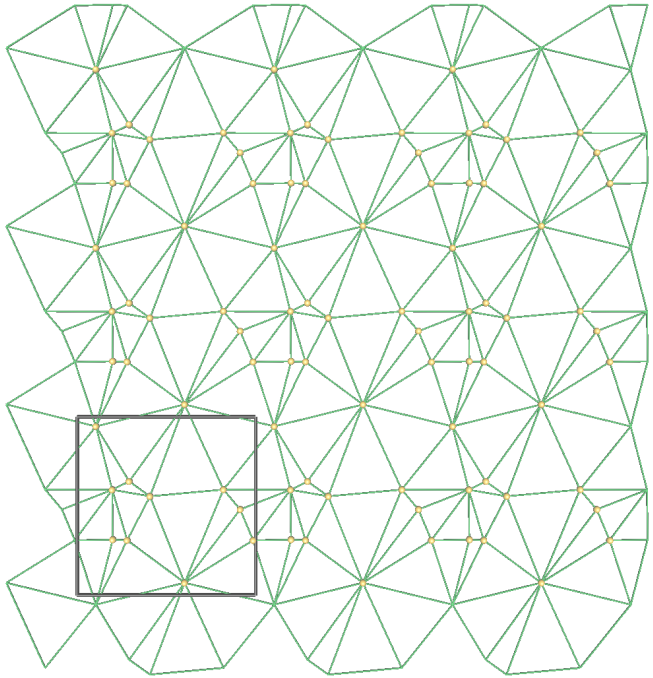
- Geometric iterators
- Parameter `Iterator_type it`



`it == AS_STORED`

Access II

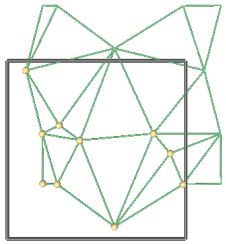
- Geometric iterators
- Parameter `Iterator_type it`



`it == COVER_DOMAIN`

Access II

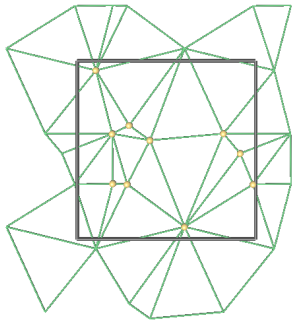
- Geometric iterators
- Parameter `Iterator_type it`



`it == UNIQUE`

Access II

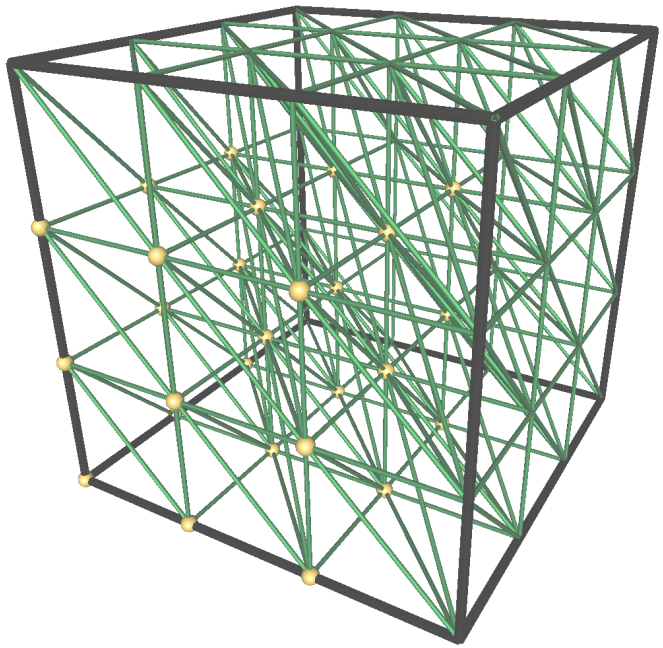
- Geometric iterators
- Parameter `Iterator_type it`



`it == COVER_DOMAIN_UNIQUE`

Heuristic for large point sets

- insert dummy point set
- compute triangulation in 1-sheeted covering
- remove dummy points



$\mathbb{R}^3$ vs. $\mathbb{T}^3$

Delaunay triangulation ($\mathbb{R}^3$ vs. $\mathbb{T}^3$):

No. of points	$\mathbb{R}^3$	$\mathbb{T}^3$	factor	old
1000	0.015	0.022	1.47	1.62
10000	0.12	0.21	1.75	1.53
100000	1.3	2.1	1.61	1.42
1000000	15	23	1.53	1.31

all times in seconds

Computing in 1-sheeted covering in $\mathbb{T}^3$.

Machine: MacBook Pro
Processor: 2.33 GHz Intel Core 2 Duo

Outline

Demo

Conclusion and Outlook

- Delaunay triangulation in $\mathbb{T}^3$
- Implementation in CGAL.
- Demo
- Improve implementation
- Non-cubic fundamental domain
- More features?

Conclusion and Outlook

- Delaunay triangulation in $\mathbb{T}^3$
- Implementation in CGAL.
- Demo
- Improve implementation
- Non-cubic fundamental domain
- More features?

Thank you
for your attention!