Convex and concave decomposition of digitized shapes using plane probing and visibility

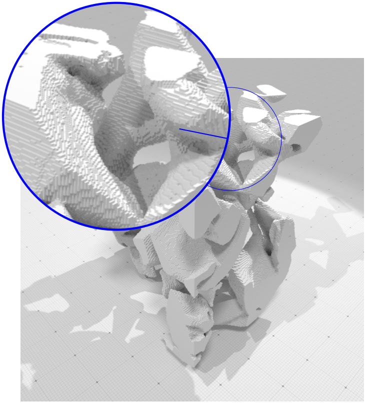
Jacques-Olivier Lachaud[1], Tristan Roussillon[2]

[1]LAMA, University Savoie Mont Blanc
[2]LIRIS, INSA Lyon

November 4th, 2025
Discrete Geometry and Mathematical Morphology (DGMM 2025)
University of Groningen, the Netherlands

Context: geometry of shapes in 3D imaging

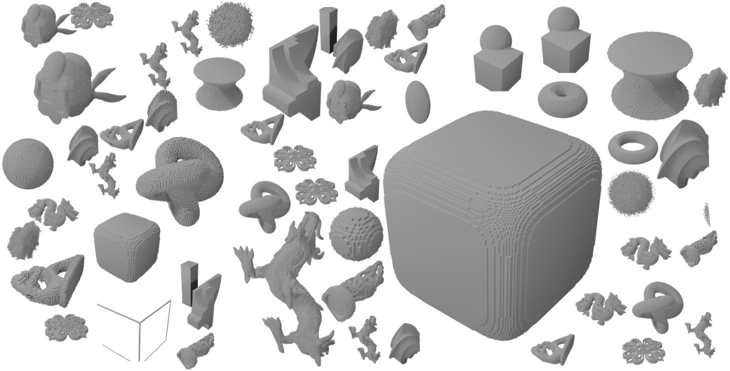
MRI, CT-scan, PET-scan, confocal microscopy, . . .



snow micro-tomography

Context: geometry of shapes in 3D imaging

geometric modeling, shape indexing, machine learning, . . .

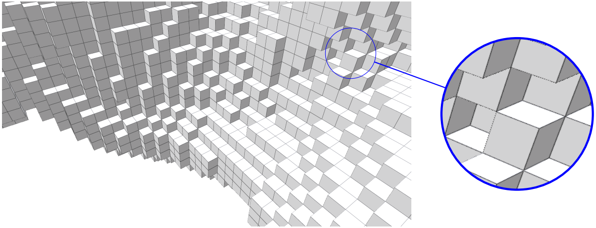


(https://github.com/dcoeurjo/VolGallery)

Digitized shapes and surfaces

Definitions

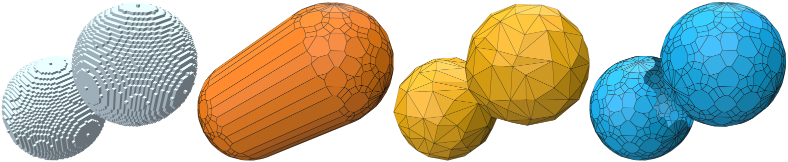
Digitized shape Z = set of *voxels*, i.e., unit cubes
Digitized surface ∂Z = boundary of Z, set of unit squares



Properties of digitized surfaces

topology	closed, oriented, but non manifold in general
geometry	approximate positions, integer points (arithmetic), uniform density, few normals

How to identify/represent convex and concave parts?



2 balls $B_{25} \cap \mathbb{Z}^3$ the convex hull an approximation local convexity

- identify vertices that are locally extremal in some direction
- identify edges and faces joining them
  - edges should form convex angles, faces around vertices should form convex cones
  - edges and faces should stay close to the digitized surface, without crossing it

Outline

Definition of local convexity/concavity based on visibility

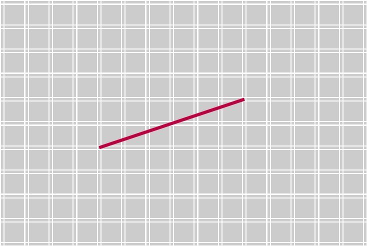
Algorithm to reconstruct convex/concave parts

Framework

Cubical grid

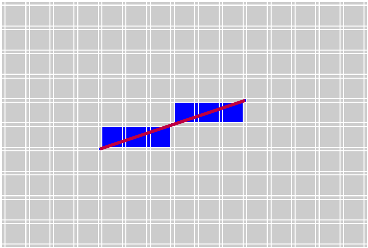
- cubical grid $\mathcal{C}^d$: partition of $\mathbb{R}^d$, where every cell is a cartesian product of d intervals of the form $\{x\}$ (closed) or $(x, x+1)$ (open)
- the dimension of a cell is the number of open intervals
- $\mathcal{C}_k^d$ denotes the set of k-dimensional cells (k-cells), $\mathcal{C}_0^d = \mathbb{Z}^d$

Basic topological notions



Cover and star of $Y \subset \mathbb{R}^d$

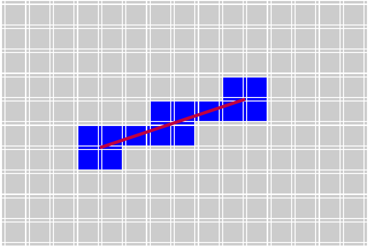
Basic topological notions



Cover and star of $Y \subset \mathbb{R}^d$

- $\mathrm{Cover}(Y) := \{c \in \mathcal{C}^d, c \cap Y \neq \emptyset\}$,

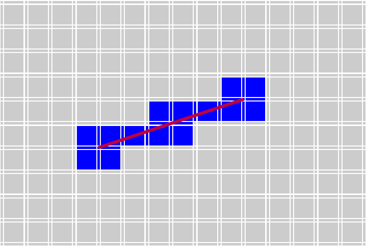
Basic topological notions



Cover and star of $Y \subset \mathbb{R}^d$

- $\mathrm{Cover}(Y) := \{c \in \mathcal{C}^d, c \cap Y \neq \emptyset\}$,
- $\mathrm{Star}(Y) := \{c \in \mathcal{C}^d, \bar{c} \cap Y \neq \emptyset\}$

Basic topological notions

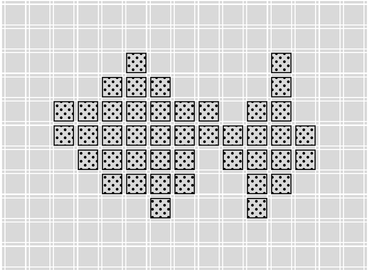


Cover and star of $Y \subset \mathbb{R}^d$

- $\mathrm{Cover}(Y) := \{c \in \mathcal{C}^d, c \cap Y \neq \emptyset\}$,
- $\mathrm{Star}(Y) := \{c \in \mathcal{C}^d, \bar{c} \cap Y \neq \emptyset\}$

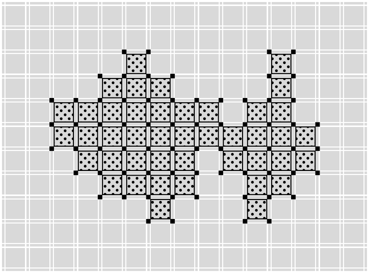
We have $\mathrm{Cover}(Y) \subseteq \mathrm{Star}(Y)$.

Basic definitions



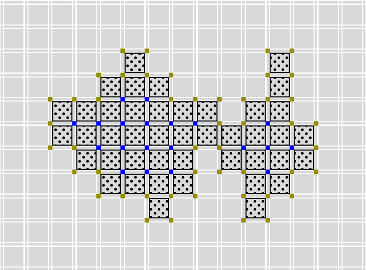
- Input: set of d-cells K

Basic definitions



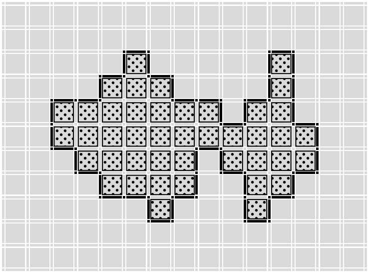
- Input: set of d-cells K
- X_K the 0-cells of the closure of K

Basic definitions



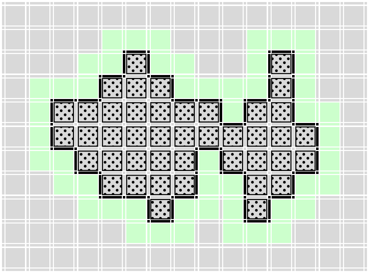
- Input: set of d-cells K
- X_K the 0-cells of the closure of K
- $X_K = \underbrace{I_K}_{\text{inner}} \sqcup \underbrace{Z_K}_{\text{boundary}}$

Basic definitions



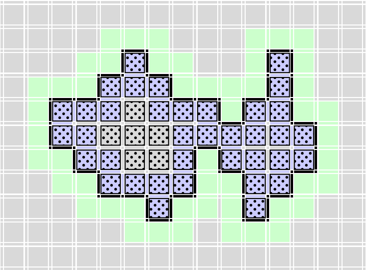
- Input: set of d-cells K
- X_K the 0-cells of the closure of K
- $X_K = \underbrace{I_K}_{\text{inner}} \sqcup \underbrace{Z_K}_{\text{boundary}}$
- $\mathrm{Bd}\,(K) := \{c \in \mathcal{C}^d_{\leqslant d-1}, \mathrm{Extr}\,(c) \subset Z_K\}$

Basic definitions



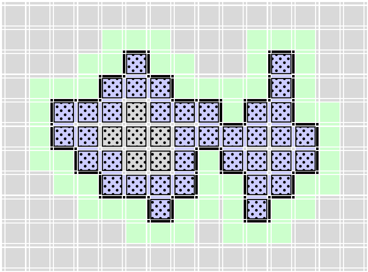
- Input: set of d-cells K
- X_K the 0-cells of the closure of K
- $X_K = \underbrace{I_K}_{\text{inner}} \sqcup \underbrace{Z_K}_{\text{boundary}}$
- $\mathrm{Bd}\,(K) := \{c \in \mathcal{C}^d_{\leqslant d-1}, \mathrm{Extr}\,(c) \subset Z_K\}$
- $\mathrm{Out}\,(K) := \{c \in \mathcal{C}^d, \mathrm{Extr}\,(c) \cap (\mathbb{Z}^d \setminus X_K) \neq \emptyset \text{ and } \mathrm{Extr}\,(c) \cap Z_K \neq \emptyset\}$

Basic definitions



- Input: set of d-cells K
- X_K the 0-cells of the closure of K
- $X_K = \underbrace{I_K}_{\text{inner}} \sqcup \underbrace{Z_K}_{\text{boundary}}$
- $\mathrm{Bd}\,(K) := \{c \in \mathcal{C}^d_{\leqslant d-1}, \mathrm{Extr}\,(c) \subset Z_K\}$
- $\mathrm{Out}\,(K) := \{c \in \mathcal{C}^d, \mathrm{Extr}\,(c) \cap (\mathbb{Z}^d \setminus X_K) \neq \emptyset$ and $\mathrm{Extr}\,(c) \cap Z_K \neq \emptyset\}$
- $\mathrm{In}\,(K) := \{c \in \mathcal{C}^d, \mathrm{Extr}\,(c) \cap Z_K \neq \emptyset$ and $\mathrm{Extr}\,(c) \cap I_K \neq \emptyset\}$

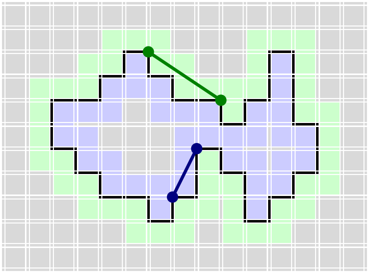
Basic definitions



- Input: set of d-cells K
- X_K the 0-cells of the closure of K
- $X_K = \underbrace{I_K}_{\text{inner}} \sqcup \underbrace{Z_K}_{\text{boundary}}$
- $\mathrm{Bd}(K) := \{c \in \mathcal{C}^d_{\leqslant d-1}, \mathrm{Extr}(c) \subset Z_K\}$
- $\mathrm{Out}(K) := \{c \in \mathcal{C}^d, \mathrm{Extr}(c) \cap (\mathbb{Z}^d \setminus X_K) \neq \emptyset \text{ and } \mathrm{Extr}(c) \cap Z_K \neq \emptyset\}$
- $\mathrm{In}(K) := \{c \in \mathcal{C}^d, \mathrm{Extr}(c) \cap Z_K \neq \emptyset \text{ and } \mathrm{Extr}(c) \cap I_K \neq \emptyset\}$

We have $\mathrm{Star}(\mathrm{Bd}(K)) = \mathrm{Bd}(K) \sqcup \mathrm{Out}(K) \sqcup \mathrm{In}(K)$

Convex and concave visibility



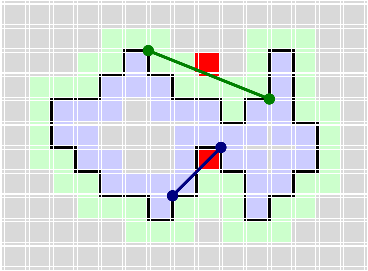
Convex K-visibility

$A := \{p_1, \ldots, p_n\} \subset Z_K$ is **convex K-visible** iff $\mathrm{Cover}\,(\mathrm{Cvxh}\,(A)) \subset \mathrm{Out}\,(K) \cup \mathrm{Bd}\,(K)$

Concave K-visibility

$A := \{p_1, \ldots, p_n\} \subset Z_K$ is **concave K-visible** iff $\mathrm{Cover}\,(\mathrm{Cvxh}\,(A)) \subset \mathrm{In}\,(K) \cup \mathrm{Bd}\,(K)$

Convex and concave visibility



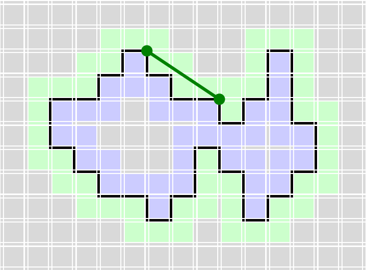
Convex K-visibility

$A := \{p_1, \ldots, p_n\} \subset Z_K$ is **convex K-visible** iff $\mathrm{Cover}(\mathrm{Cvxh}(A)) \subset \mathrm{Out}(K) \cup \mathrm{Bd}(K)$

Concave K-visibility

$A := \{p_1, \ldots, p_n\} \subset Z_K$ is **concave K-visible** iff $\mathrm{Cover}(\mathrm{Cvxh}(A)) \subset \mathrm{In}(K) \cup \mathrm{Bd}(K)$

Convex and concave visibility



Convex K-visibility

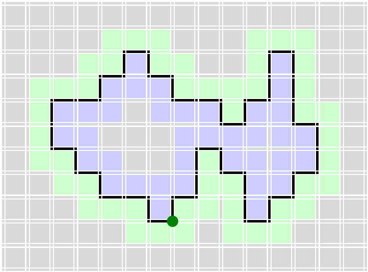
$A := \{p_1, \ldots, p_n\} \subset Z_K$ is **convex K-visible** iff $\mathrm{Cover}\,(\mathrm{Cvxh}\,(A)) \subset \mathrm{Out}\,(K) \cup \mathrm{Bd}\,(K)$

Concave K-visibility

$A := \{p_1, \ldots, p_n\} \subset Z_K$ is **concave K-visible** iff $\mathrm{Cover}\,(\mathrm{Cvxh}\,(A)) \subset \mathrm{In}\,(K) \cup \mathrm{Bd}\,(K)$

From now on, focus on convex visibility (concave visibility is entirely symmetric).

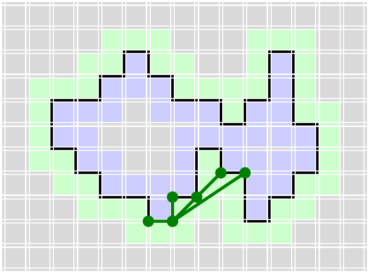
K-visibility cone, locally-convex point, edge, face, ...



$C_K(p)$

The **convex** K**-visibility cone** $C_K(p)$ of p is the set of points $q \in Z_K$ with $\{p, q\}$ convex K-visible.

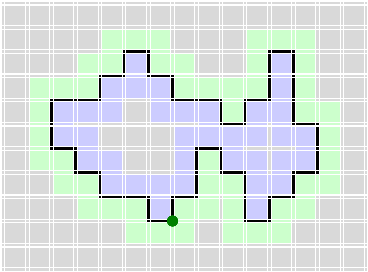
K-visibility cone, locally-convex point, edge, face, ...



$C_K(p)$

The **convex K-visibility cone** $C_K(p)$ of p is the set of points $q \in Z_K$ with $\{p, q\}$ convex K-visible.

K-visibility cone, locally-convex point, edge, face, ...



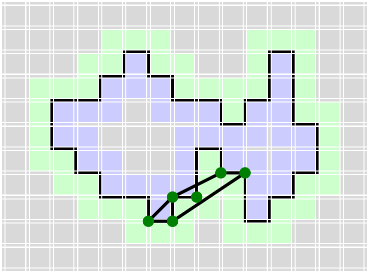
$C_K(p)$

The **convex K-visibility cone** $C_K(p)$ of p is the set of points $q \in Z_K$ with $\{p, q\}$ convex K-visible.

Locally convex point

Point $p \in Z_K$ is **locally convex** in K iff it is a vertex of $\mathrm{Cvxh}\,(C_K(p))$

K-visibility cone, locally-convex point, edge, face, ...



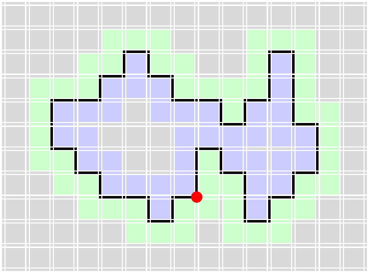
$C_K(p)$

The **convex K-visibility cone** $C_K(p)$ of p is the set of points $q \in Z_K$ with $\{p, q\}$ convex K-visible.

Locally convex point

Point $p \in Z_K$ is **locally convex** in K iff it is a vertex of $\mathrm{Cvxh}\,(C_K(p))$

K-visibility cone, locally-convex point, edge, face, ...



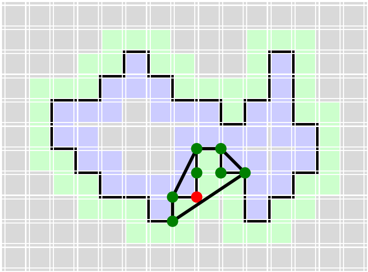
$C_K(p)$

The **convex K-visibility cone** $C_K(p)$ of p is the set of points $q \in Z_K$ with $\{p, q\}$ convex K-visible.

Locally convex point

Point $p \in Z_K$ is **locally convex** in K iff it is a vertex of $\mathrm{Cvxh}\left(C_K(p)\right)$

K-visibility cone, locally-convex point, edge, face, ...



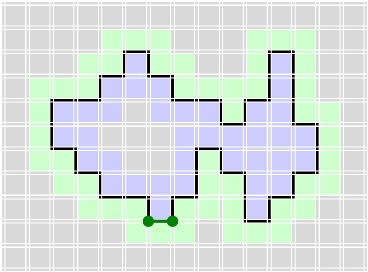
$C_K(p)$

The **convex K-visibility cone** $C_K(p)$ of p is the set of points $q \in Z_K$ with $\{p, q\}$ convex K-visible.

Locally convex point

Point $p \in Z_K$ is **locally convex** in K iff it is a vertex of $\mathrm{Cvxh}\,(C_K(p))$

K-visibility cone, locally-convex point, edge, face, ...



$C_K(p)$

The **convex K-visibility cone** $C_K(p)$ of p is the set of points $q \in Z_K$ with $\{p, q\}$ convex K-visible.

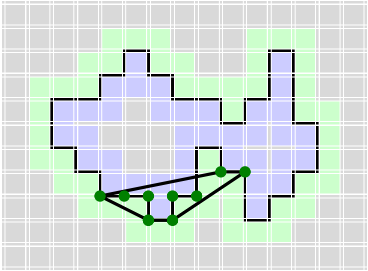
Locally convex point

Point $p \in Z_K$ is **locally convex** in K iff it is a vertex of $\mathrm{Cvxh}\left(C_K(p)\right)$

Locally convex edge, face, ...

Face $\{p_i\} \subset Z_K$ is **locally convex** in K iff it is a face of $\mathrm{Cvxh}\left(\cup_{p_i} C_K(p_i)\right)$.

K-visibility cone, locally-convex point, edge, face, ...



$C_K(p)$

The **convex K-visibility cone** $C_K(p)$ of p is the set of points $q \in Z_K$ with $\{p, q\}$ convex K-visible.

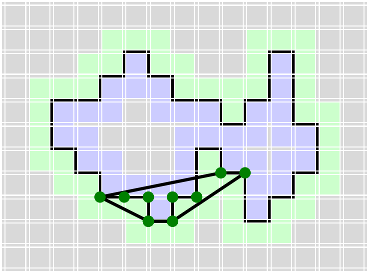
Locally convex point

Point $p \in Z_K$ is **locally convex** in K iff it is a vertex of $\mathrm{Cvxh}\,(C_K(p))$

Locally convex edge, face, ...

Face $\{p_i\} \subset Z_K$ is **locally convex** in K iff it is a face of $\mathrm{Cvxh}\,(\cup_{p_i} C_K(p_i))$.

K-visibility cone, locally-convex point, edge, face, ...



$C_K(p)$

The **convex K-visibility cone** $C_K(p)$ of p is the set of points $q \in Z_K$ with $\{p, q\}$ convex K-visible.

Locally convex point

Point $p \in Z_K$ is **locally convex** in K iff it is a vertex of $\mathrm{Cvxh}\,(C_K(p))$

Locally convex edge, face, ...

Face $\{p_i\} \subset Z_K$ is **locally convex** in K iff it is a face of $\mathrm{Cvxh}\,(\cup_{p_i} C_K(p_i))$.

Lemma (Consistency of local convexity)

If F is locally convex in K, then any subset of F is locally convex in K.

Full convexity implies local convexity

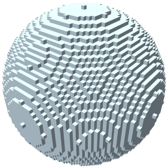
Full convexity [Lachaud. 2021]

The digital set $X \subset \mathbb{Z}^d$ is fully convex iff $\mathrm{Star}\,(\mathrm{Cvxh}\,(X)) \subset \mathrm{Star}\,(X)$.

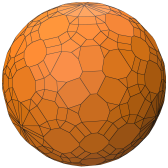
- full convexity implies classical digital convexity
- full convexity implies connectedness

Theorem

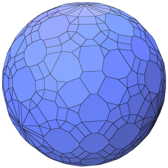
Let $K \subset \mathcal{C}^d_d$ and X_K fully convex. The vertices and the faces of $\mathrm{Cvxh}\,(X_K)$ are locally convex vertices and locally convex faces of K.



input K



$\mathrm{Cvxh}\,(X_K)$



locally convex faces

Outline

Definition of local convexity/concavity based on visibility

Algorithm to reconstruct convex/concave parts

Algorithm to reconstruct convex parts

Input a set of d-dimensional cells K

- compute the boundary 0-cells Z_K of K
- compute the visibility cones $C_K(p)$, for all $p \in Z_K$
- for all point $p \in Z_K$
  - check if p is locally convex by computing $\mathrm{Cvxh}\left(C_K(p)\right)$
  - collect incident edges in E if it is the case
  - store p in V if it is the case
- for all edge $e := (p_1, p_2) \in E$
  - check if e is locally convex by computing $\mathrm{Cvxh}\left(\cup_i C_K(p_i)\right)$
  - collect incident faces in F if it is the case
- for all face $f := (p_1, \ldots, p_k) \in F$
  - check if f is locally convex by computing $\mathrm{Cvxh}\left(\cup_i C_K(p_i)\right)$
  - and store it in G if it is the case
- return locally convex points V and faces G

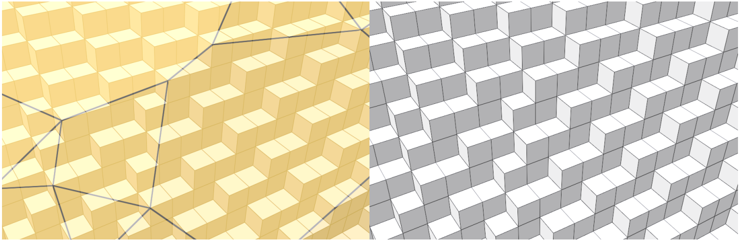
Algorithm to reconstruct convex parts

Input a set of d-dimensional cells K

- compute the boundary 0-cells Z_K of K
- compute the visibility cones $C_K(p)$, for all $p \in Z_K$
- for all point $p \in Z_K$
  - check if p is locally convex by computing $\mathrm{Cvxh}\left(C_K(p)\right)$
  - collect incident edges in E if it is the case
  - store p in V if it is the case
- for all edge $e := (p_1, p_2) \in E$
  - check if e is locally convex by computing $\mathrm{Cvxh}\left(\cup_i C_K(p_i)\right)$
  - collect incident faces in F if it is the case
- for all face $f := (p_1, \ldots, p_k) \in F$
  - check if f is locally convex by computing $\mathrm{Cvxh}\left(\cup_i C_K(p_i)\right)$
  - and store it in G if it is the case
- return locally convex points V and faces G

More than 95% of the time is spent in computing visibility cones.
$\Rightarrow$ We have to prune the input set Z_K.

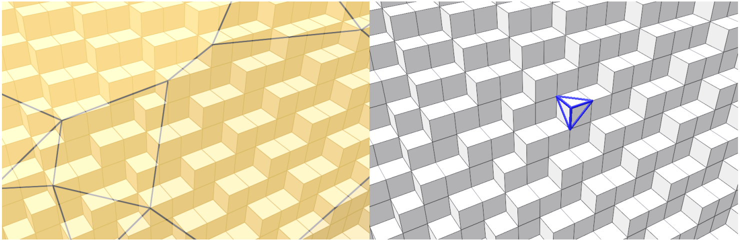
Discarding non-extremal points



Objective

Quickly discard points of Z_K that cannot be locally convex points, without computing their visibility cone.

Discarding non-extremal points

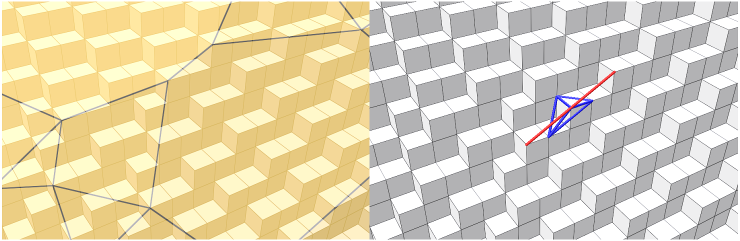


Objective

Quickly discard points of Z_K that cannot be locally convex points, without computing their visibility cone.

- keep only the salient corners of Z_K

Discarding non-extremal points

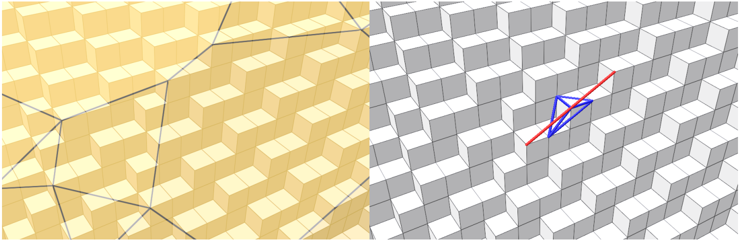


Objective

Quickly discard points of Z_K that cannot be locally convex points, without computing their visibility cone.

- keep only the salient corners of Z_K
- eliminate corners c that are in-between two points of Z_K, i.e., $\exists \mathbf{v} \in \mathbb{Z}^d$ with $c \pm \mathbf{v} \in Z_K$

Discarding non-extremal points

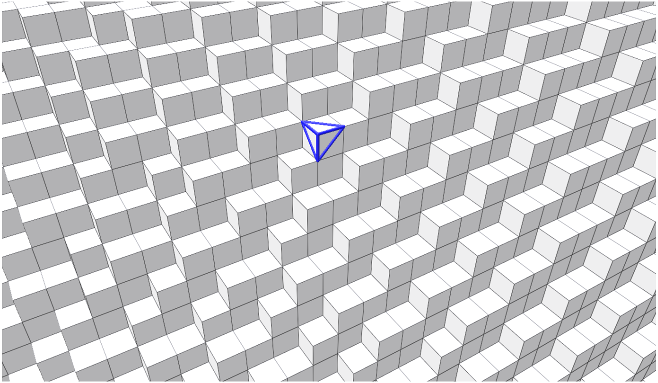


Objective

Quickly discard points of Z_K that cannot be locally convex points, without computing their visibility cone.

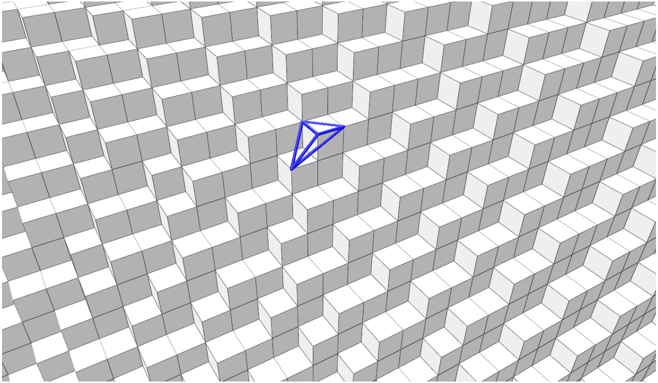
- keep only the salient corners of Z_K
- eliminate corners c that are in-between two points of Z_K, i.e., $\exists \mathbf{v} \in \mathbb{Z}^d$ with $c \pm \mathbf{v} \in Z_K$
- method to find $\mathbf{v}$: a variant of plane probing, normally used for plane recognition [Lauchaud, Provençal, Roussillon. 17]

Illustration of our algorithm IsExtremal



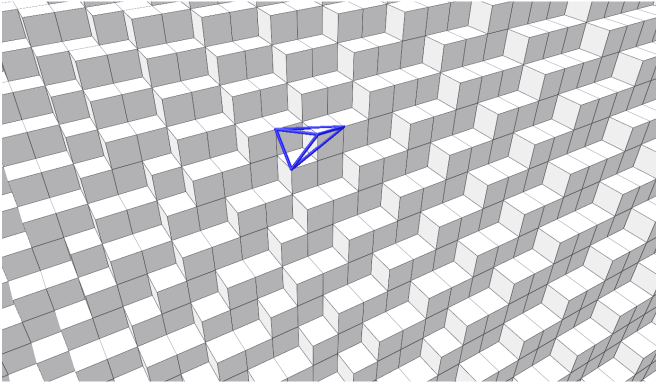
- Tetrahedron with one vertex fixed (the point to test), described by a matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$
- Initialize $\mathbf{M}$ with $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and update it

Illustration of our algorithm IsExtremal



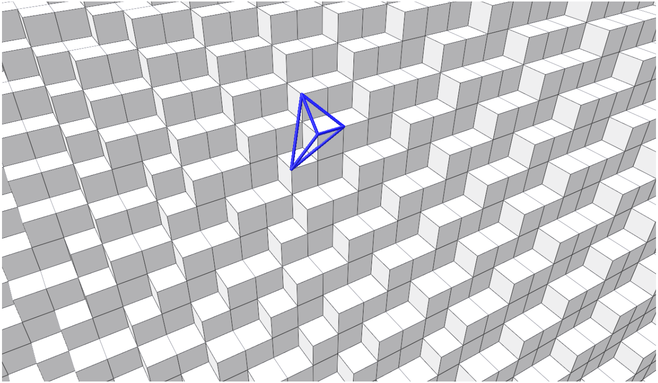
- Tetrahedron with one vertex fixed (the point to test), described by a matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$
- Initialize $\mathbf{M}$ with $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and update it

Illustration of our algorithm IsExtremal



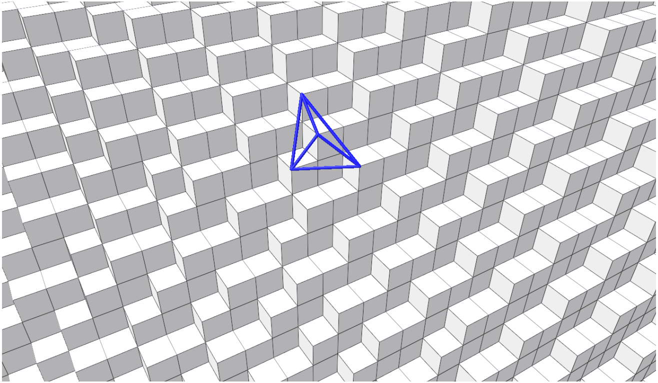
- Tetrahedron with one vertex fixed (the point to test), described by a matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$
- Initialize $\mathbf{M}$ with $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and update it

Illustration of our algorithm IsExtremal



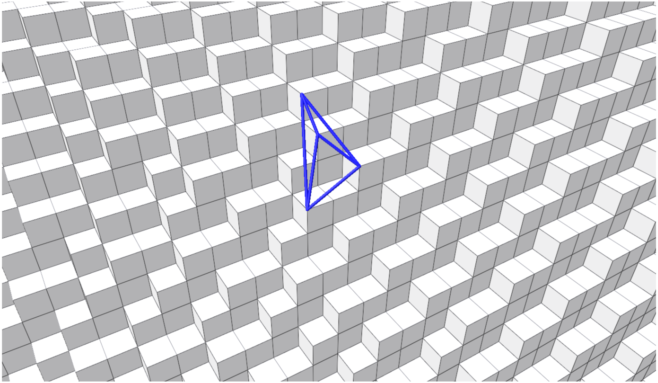
- Tetrahedron with one vertex fixed (the point to test), described by a matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$
- Initialize $\mathbf{M}$ with $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and update it

Illustration of our algorithm IsExtremal



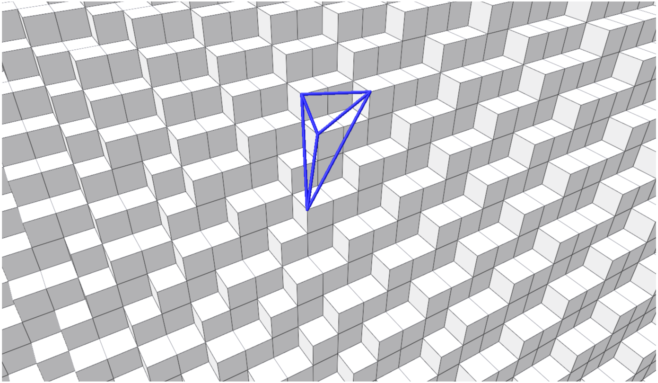
- Tetrahedron with one vertex fixed (the point to test), described by a matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$
- Initialize $\mathbf{M}$ with $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and update it

Illustration of our algorithm IsExtremal



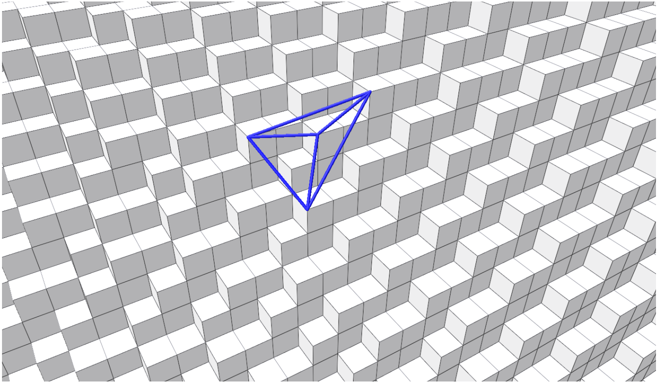
- Tetrahedron with one vertex fixed (the point to test), described by a matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$
- Initialize $\mathbf{M}$ with $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and update it

Illustration of our algorithm IsExtremal



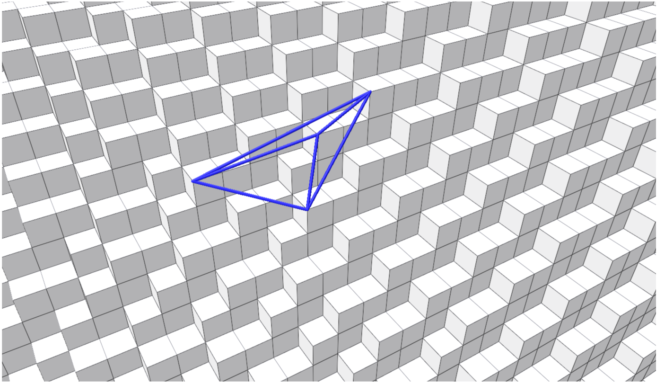
- Tetrahedron with one vertex fixed (the point to test), described by a matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$
- Initialize $\mathbf{M}$ with $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and update it

Illustration of our algorithm IsExtremal



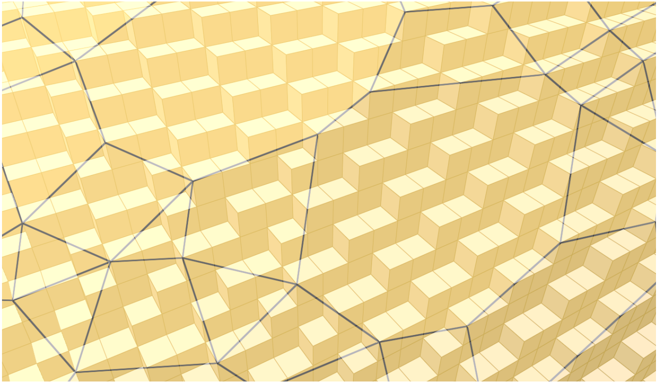
- Tetrahedron with one vertex fixed (the point to test), described by a matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$
- Initialize $\mathbf{M}$ with $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and update it

Illustration of our algorithm IsExtremal



- Tetrahedron with one vertex fixed (the point to test), described by a matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$
- Initialize $\mathbf{M}$ with $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and update it

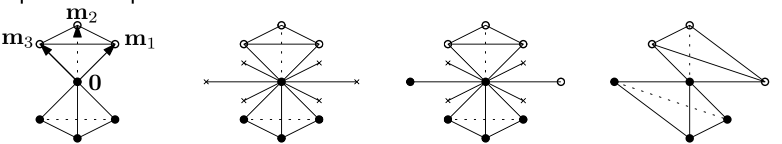
Illustration of our algorithm IsExtremal



- Tetrahedron with one vertex fixed (the point to test), described by a matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$
- Initialize $\mathbf{M}$ with $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and update it

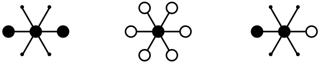
Algorithm IsExtremal

- Invariant: $\det(\mathbf{M}) = 1$, $-\mathbf{m}_k \in Z_K, \mathbf{m}_k \notin Z_K$
- Loop these steps:



vectors $\mathbf{m}_k$ go up, $\mathbf{m}_k - \mathbf{m}_{k\pm1}$ in Z_K? if yes, $\mathbf{m}_k \leftarrow \mathbf{m}_k - \mathbf{m}_{k\pm1}$

- possible configurations at the six points $\mathbf{m}_k - \mathbf{m}_{k\pm1}$:



NO YES PROBING

Algorithm termination

Theorem

If Z_k is finite, algorithm $\textsc{IsExtremal}$ *terminates after a finite number of iterations.*

Theorem

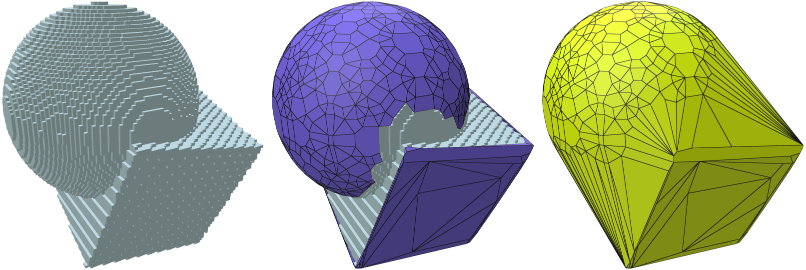
If the point to test $\mathbf{x}$ *is a vertex of* $\mathrm{Cvxh}\left(Z_k\right)$*, algorithm* $\textsc{IsExtremal}$ *returns* $\textsc{Yes}$ *after at most n iterations, with $n \leqslant 2\sqrt{3}A$ and A the total area of the facets of* $\mathrm{Cvxh}\left(Z_K\right)$ *incident to* $\mathbf{x}$.

How good is the probing algorithm as a filter?

- efficient on digitizations of smooth shapes (here ellipsoid) with gridstep h
- n_{init}: number of salient corners
- n_{final}: corners labeled as extremal by algorithm IsExtremal
- $n_{\mathrm{Cvxh}(Z_K)}$: expected number of vertices of $\mathrm{Cvxh}(Z_K)$

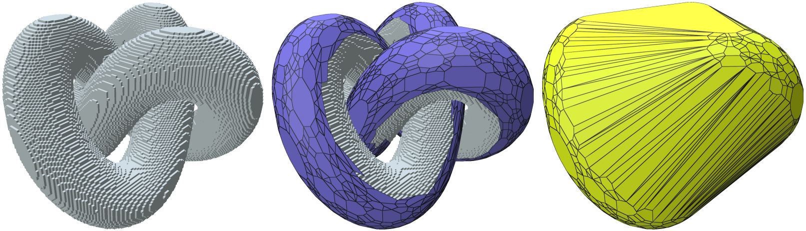
grid step	$\#Z_K$	n_{init}	n_{final}	$n_{\mathrm{Cvxh}(Z_K)}$
0.5	984	112	112	112
0.1	24.808	2.032	1.128	1.128
0.05	99.448	7.784	3.064	3.064
0.01	2.488.104	186.664	33.864	33.784

A few results (convex zones)



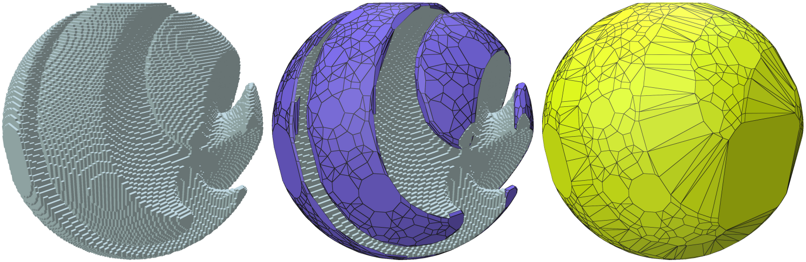
shape	$\#Z_K$	n_{init}	n_{final}	#facets	time(ms)
cps	34036	3681	991	959	2529
torus-knot-128	96622	15196	2924	2752	29321
sharpsphere129	119846	16715	3099	2542	40492

A few results (convex zones)



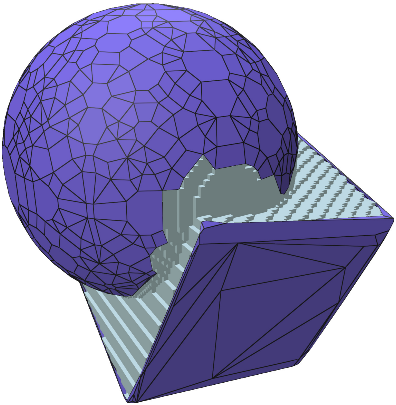
shape	$\#Z_K$	n_{init}	n_{final}	#facets	time(ms)
cps	34036	3681	991	959	2529
torus-knot-128	96622	15196	2924	2752	29321
sharpsphere129	119846	16715	3099	2542	40492

A few results (convex zones)

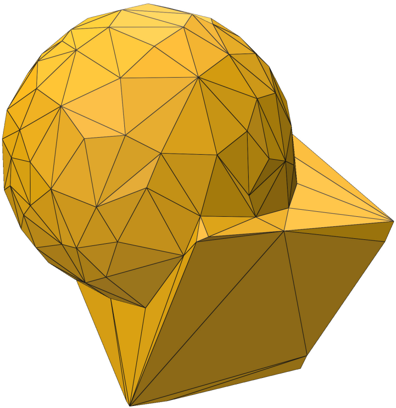


shape	$\#Z_K$	n_{init}	n_{final}	#facets	time(ms)
cps	34036	3681	991	959	2529
torus-knot-128	96622	15196	2924	2752	29321
sharpsphere129	119846	16715	3099	2542	40492

Comparison with greedy triangulation



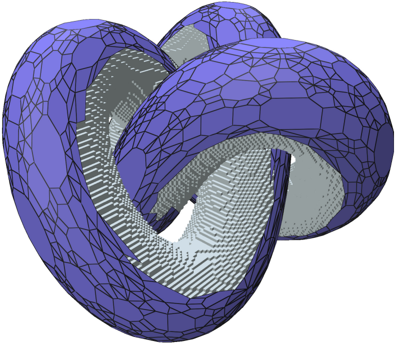
our method



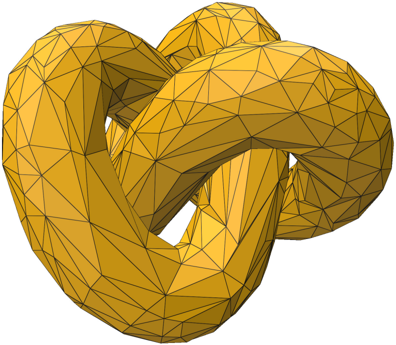
greedy triangulation

Both methods are at Hausdorff distance 1 from digital surface.

Comparison with greedy triangulation



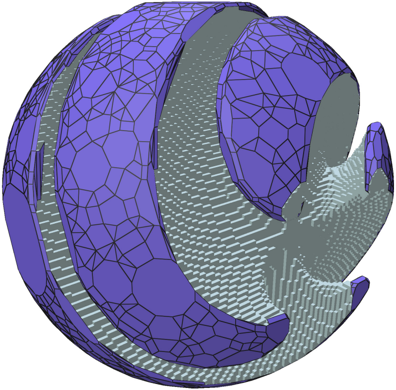
our method



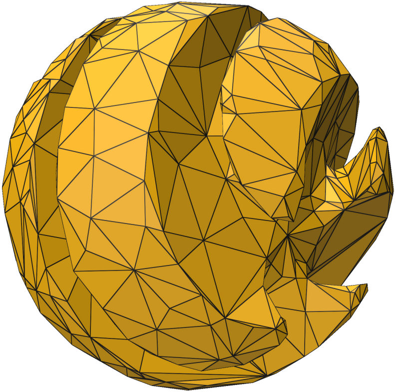
greedy triangulation

Both methods are at Hausdorff distance 1 from digital surface.

Comparison with greedy triangulation



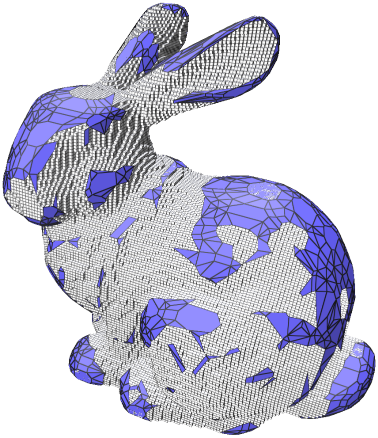
our method



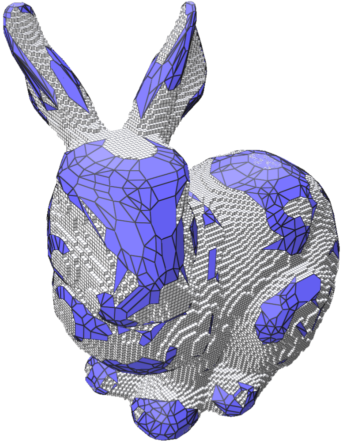
greedy triangulation

Both methods are at Hausdorff distance 1 from digital surface.

A last result (convex zones)



A last result (convex zones)



Conclusion and perspectives

Contribution

- local definition of convexity/concavity through convex hulls of visibility cones
- algorithm that reconstructs the locally convex/concave vertices, edges and faces, ...
- fast probing algorithm to identify at 99% extremal points

Perspectives

- speed up: compute visibility in a coarse-to-fine way
- How to triangulate neither convex nor concave parts?
  - use other probing variants to identify saddle vertices/edges
  - perform a greedy triangulation with constrained vertices, edges and faces given by local convexity/concavity

Conclusion and perspectives

Contribution

- local definition of convexity/concavity through convex hulls of visibility cones
- algorithm that reconstructs the locally convex/concave vertices, edges and faces, ...
- fast probing algorithm to identify at 99% extremal points

Perspectives

- speed up: compute visibility in a coarse-to-fine way
- How to triangulate neither convex nor concave parts?
  - use other probing variants to identify saddle vertices/edges
  - perform a greedy triangulation with constrained vertices, edges and faces given by local convexity/concavity

Thank you for your attention !