# An Integrated Framework for Exact and Efficient Collision Detection of Convex Polyhedra

Research Proposal

Efi Fogel

efif@post.tau.ac.il

School of Computer Science, Tel Aviv University

# Outline

- Introduction
- Warm up — 2D Collision Detection
- Infrastructure
- The Cubical Gaussian Map
- 3D Minkowski Sums
- 3D Collision Detection
- Summary
- The CGM Data-Structure

# Introduction
# Motivation

**Assembly Process** — the execution of an assembly algorithm that specifies assembly and/or disassembly operations on the sub components of a product and the ordering of these operations

- It is possible for two or more components to arrive at a state, where they are in contact with each other during the process

- An algorithm that operates in an inexact manner may occasionally result in the wrong decision, and damage the assembly process

# Introduction
# Background

- Collision Detection and Proximity Queries
  - Dobkin & Kirkpatrick hierarchy, Dobkin & Kirkpatrick
    $O(n + m)$ preprocessing time + $O(\log n \log m)$ query time
  - GJK algorithm —implicit Minkowski sum computation, Gilbert & Johnson & Keerthi, Cameron
  - LC algorithm — examines polytopes features, Lin & Canny
- Penetration Depth
  - Randomized approach, Agarwal et al.
- Minkowski Sums
  - Approximations, e.g., Varadhan & Manocha
  - Kinetic framework & Convolution, Guibas at al., Basch et al.
  - Slope diagrams, Ghosh, Bekker & Roerdink
- Libraries
  - SOLID —based on an improved GJK alg., Bergen
  - SWIFT —based on an advanced LC alg., Ehmann & Lin
  - QuickCD —uses bounding-polytopes hierarchy, Mitchell

# Introduction: Collision Detection and Proximity Queries of Convex Polyhedra

- Efficient algorithms
- Complete implementation
  - Exact results are guaranteed
  - All degenerate cases are handled
  - Competitive performance
- Implemented on top of CGAL
  - Arrangement and Polyhedral Surface packages
- Uses a dual representation — *Cubical Gaussian Map*
- An integrated framework, which consists of 3 layers:

| | |
|---|---|
| Infrastructure | 2D Arrangement data-structure & operations (overlay) |
| Representation | Cubical Gaussian Map data-structure & operations |
| Application | Simulations, Demonstrations, Tests, & Benchmarks |

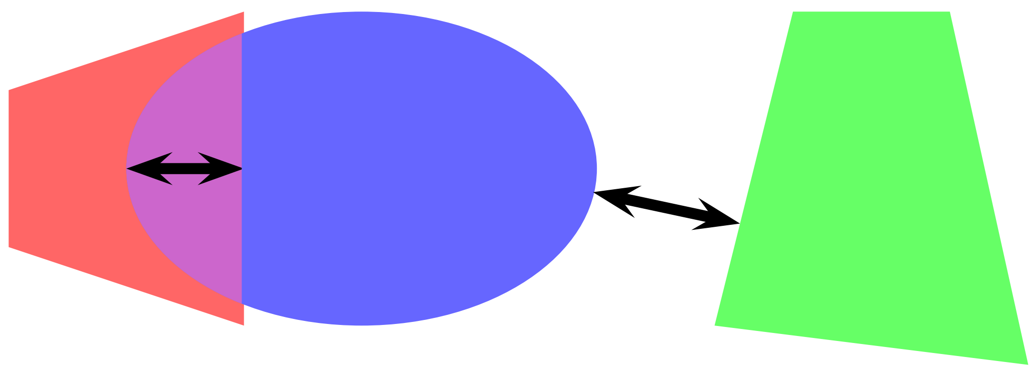
# Introduction
# Proximity Queries

- $P$ and $Q$ are two closed convex polyhedra in $\mathbb{R}^d$
  - $P$ translated by a vector $t$ is denoted by $P^t$

$\delta(P, Q) = \min\{\|t\| \mid P^t \cap Q \neq \emptyset, t \in \mathbb{R}^d\}$ separation distance

$\pi(P, Q) = \inf\{\|t\| \mid P^t \cap Q = \emptyset, t \in \mathbb{R}^d\}$ penetration depth

$\pi_d(P, Q) = \inf\{a \mid P^{\vec{da}} \cap Q = \emptyset\}$ directional penetration depth

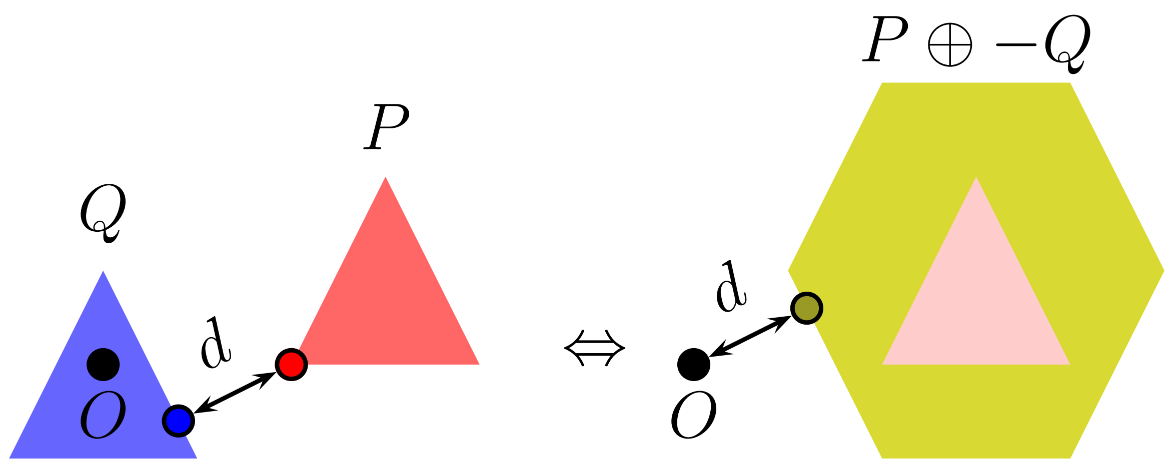


# Introduction
# Minkowski Sums

- $P$ and $Q$ are two closed convex polyhedra in $\mathbb{R}^d$

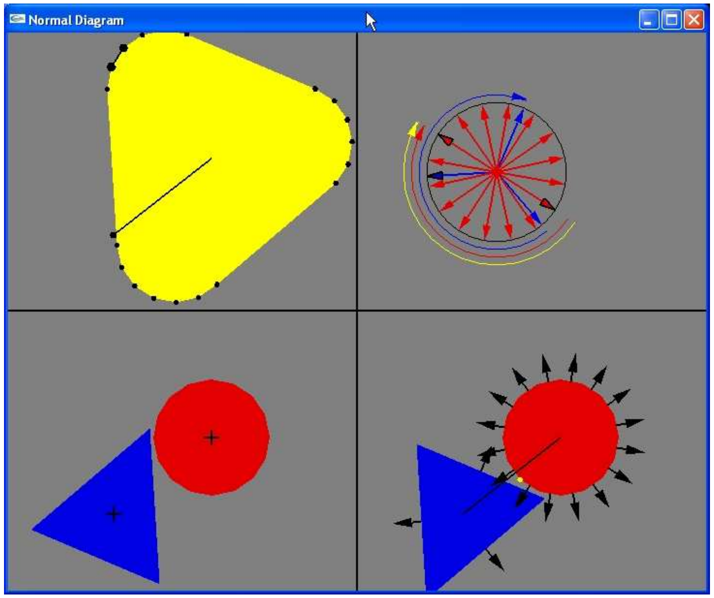
$$M = P \oplus Q = \{p + q \mid p \in P, q \in Q\}$$ Minkowski sum

The minimum separation distance between $P$ and $Q$ is the same as the minimum distance between the origin and the boundary of the Minkowski sum of $P$ and $-Q$



# Warm-up —2D Collision Detection `mink2d`

An interactive program that simulates and demonstrates 2D collision detection

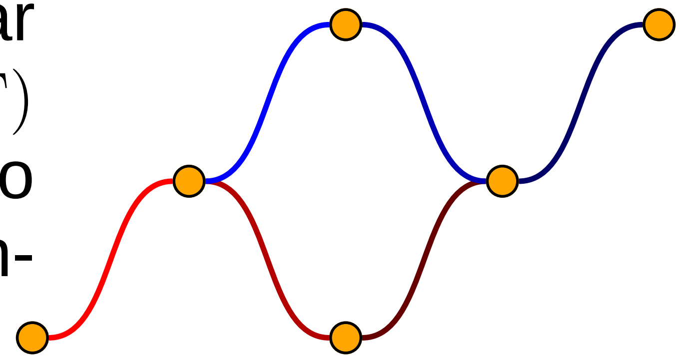


# Infrastructure

| | |
|---|---|
| Infrastructure | 2D Arrangement data-structure & operations (e.g., overlay) |
| Representation | Cubical Gaussian Map data-structure & operations |
| Application | Simulations, Demonstrations, Tests, & Benchmarks |

# Infrastructure
# Arrangement of CGAL

- Given a collection $\Gamma$ of planar curves, the arrangement $\mathcal{A}(\Gamma)$ is the partition of the plane into vertices, edges and faces induced by the curves of $\Gamma$



- CGAL — Computational Geometry Algorithms Library

**kernel** — a model of the geometric kernel concept

  - Constant-size geometric primitives (points, segments)
  - Predicates and constructors (orientation(p,q,r))

**modules** — geometric data-structures
(`Polyhedron_3`, `Triangulation_2`, `Arrangement_2`)

**miscellaneous** — windowing, I/O, circulators, etc.

# Infrastructure
# Architecture

**`Planar_map_2<Traits,Dcel>`** — maintains planar maps constructed from interior disjoint $x$-monotone curves. Topology and geometry are separated:

- **Dcel** — maintains the incidence relation on the vertices, edges, and faces
- **Traits** — defines the abstract interface between planar maps and the geometric primitives they use

**`Planar_map_with_intersections_2`** — maintains planar maps constructed from general curves (may intersect or be non-$x$-monotone)

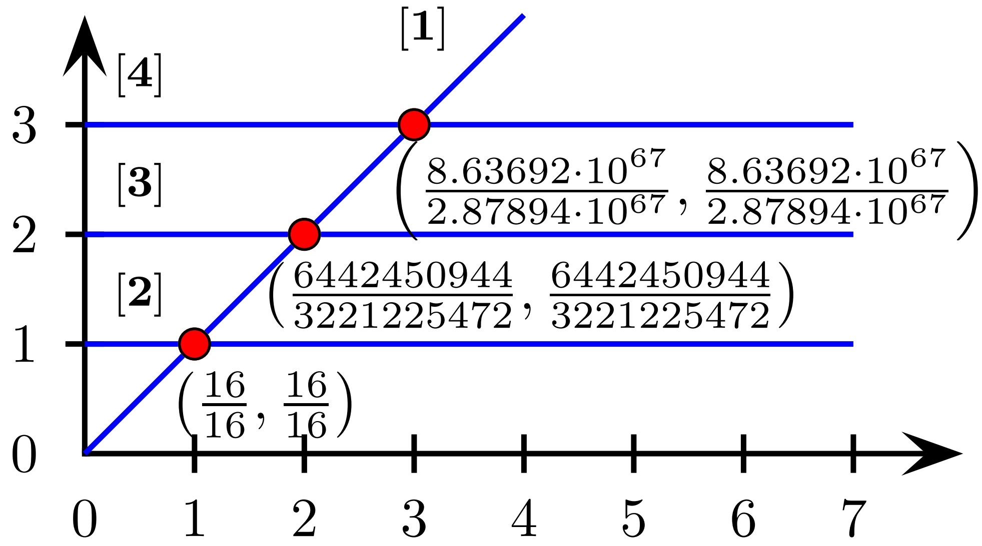
**`Arrangement_2`** — maintains planar maps of intersecting curves along with curve history

# Infrastructure
# Planar Map Traits

- Parameter of package
  - Defines the family of curves in interest
  - Package can be used with any family of curves for which a traits class is supplied
- Aggregate
  - Geometric types (point, curve)
  - Operations over types (accessors, predicates, constructors)
- Encapsulates implementation details
  - The number type used
  - The coordinate representation (Cartesian, homogeneous)
  - Extraneous data stored with the geometric objects

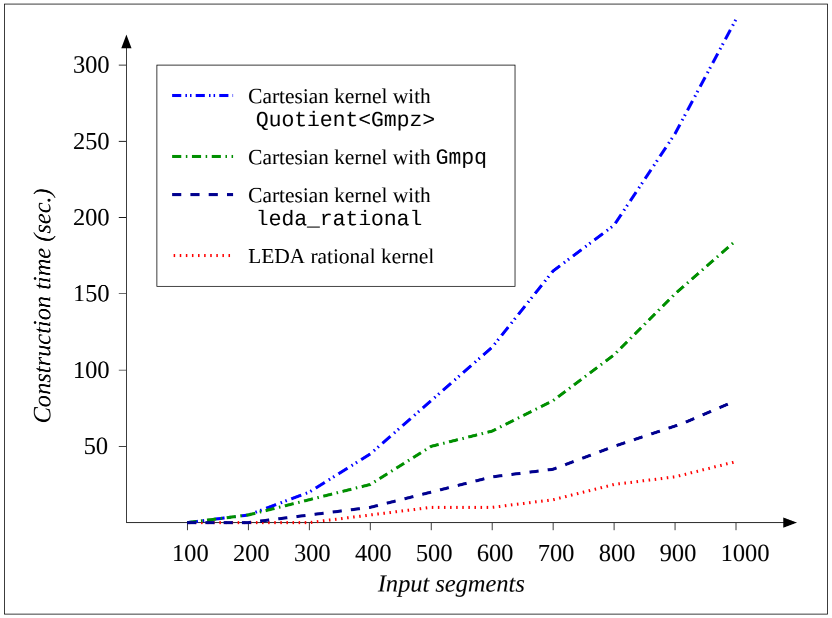
# Infrastructure
# Segment Traits

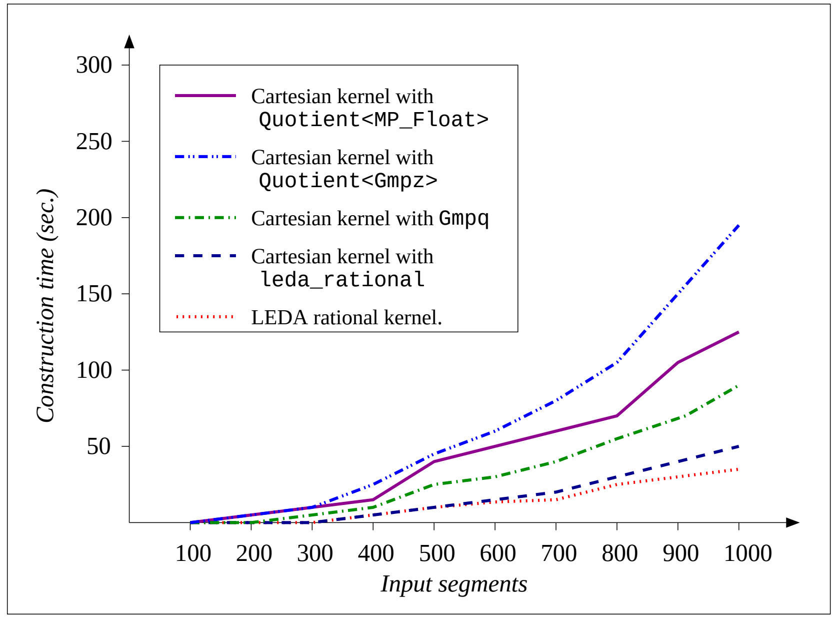
- Parameterized with a geometric kernel
- Segments reprsented only by their end points leads to cascaded representation of intersection points




- The *caching* traits avoids the problem
  - Caches information, e.g., the undelying line
  - Faster when constructing a dense arrangements
- Indiscriminate normalization can be harmful

# Infrastructure
# Caching Segment Traits




Using kernel traits

Using caching traits

- Inserting 1000 random curves with **`Quotient<Qmpz>`** is $\sim 50\%$ faster with the cached traits vs. the kernel triats
- The LEDA rational kernel uses heuristics to normalize the numbers. Therefore, the speedup is minor
- The **`Quotient<MP_Float>`** number type cannot be used at all with the kernel traits, as its precision is limited

# Infrastructure
# Map Overlay

**Red-Blue intersection** — the intersection between segements of 2 sets, each consisting of segments that are pairwise disjoint in their interiors

**Map Overlay** — the planar subdivision $\mathcal{S}$ that is the overlay of 2 planar subdivisions $\mathcal{S}_1$ and $\mathcal{S}_2$. There is a face $f$ in $\mathcal{S}$, if and only if there are faces $f_1$ and $f_2$ in $\mathcal{S}_1$ and $\mathcal{S}_2$ respectively such that $f$ is a maximal connected subset of $f_1 \cap f_2$

- A superset of the *red-blue intersection* problem
  - Requires only extra linear time
- Can be computed in $O(n \log n + k)$ optimal time
- Can be computed in $O(n + k)$, if each subdivision is simply connected

$n$ —total number of segments, $k$ —number of intersections

# The Cubical Gaussian Map

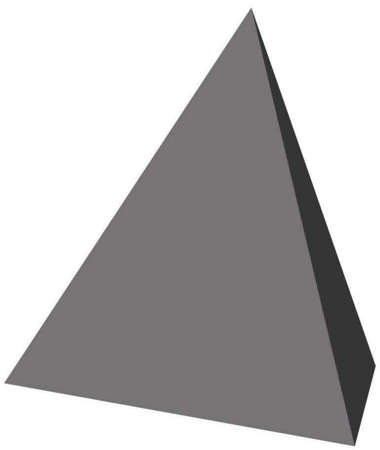
| | |
|---|---|
| Infrastructure | 2D Arrangement data-structure & operations (e.g., overlay) |
| Representation | Cubical Gaussian Map data-structure & operations |
| Application | Simulations, Demonstrations, Tests, & Benchmarks |

# The Cubical Gaussian Map
# Definitions

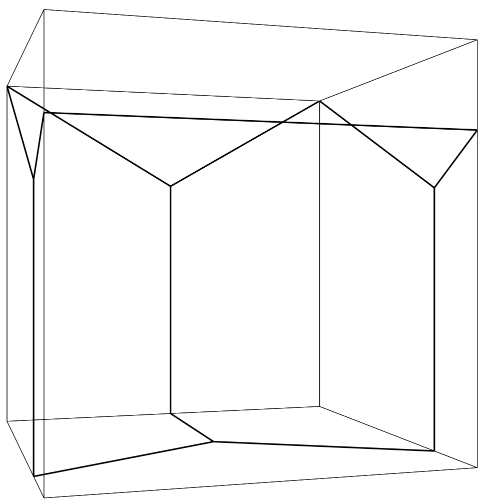
***Unit Cube*** — the parallel-axis cube circumscribing the unit sphere (its edges are of length 2)

***Cubical Gaussian Map*** — the set-valued function from a polytope $P$ to the six faces of the unit cube

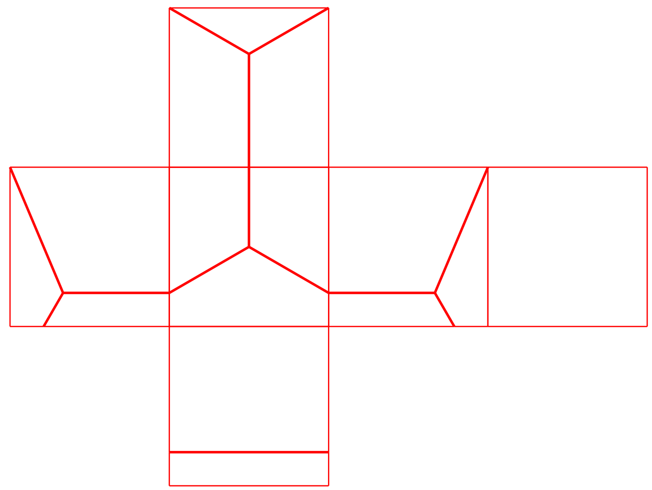
A Tetrahedron



The primal



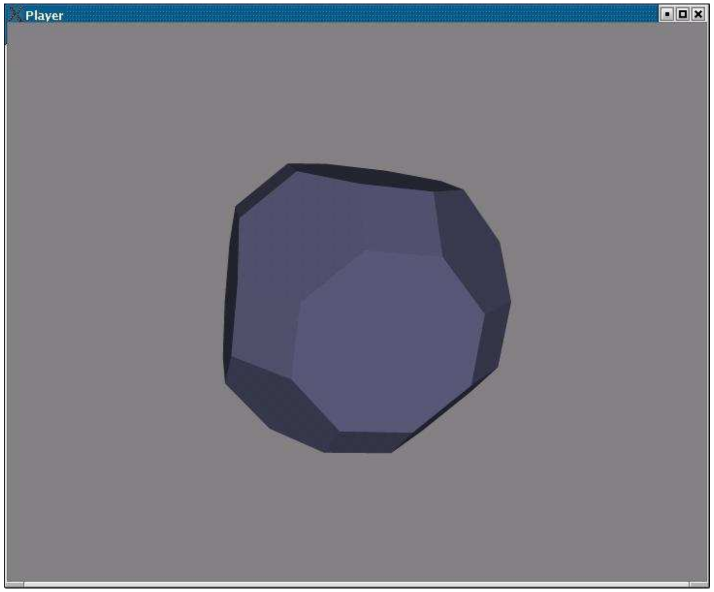
The CGM



The CGM unfolded

# The Cubical Gaussian Map `player`

An interactive program that “plays” 3D models stored in an extended VRML format



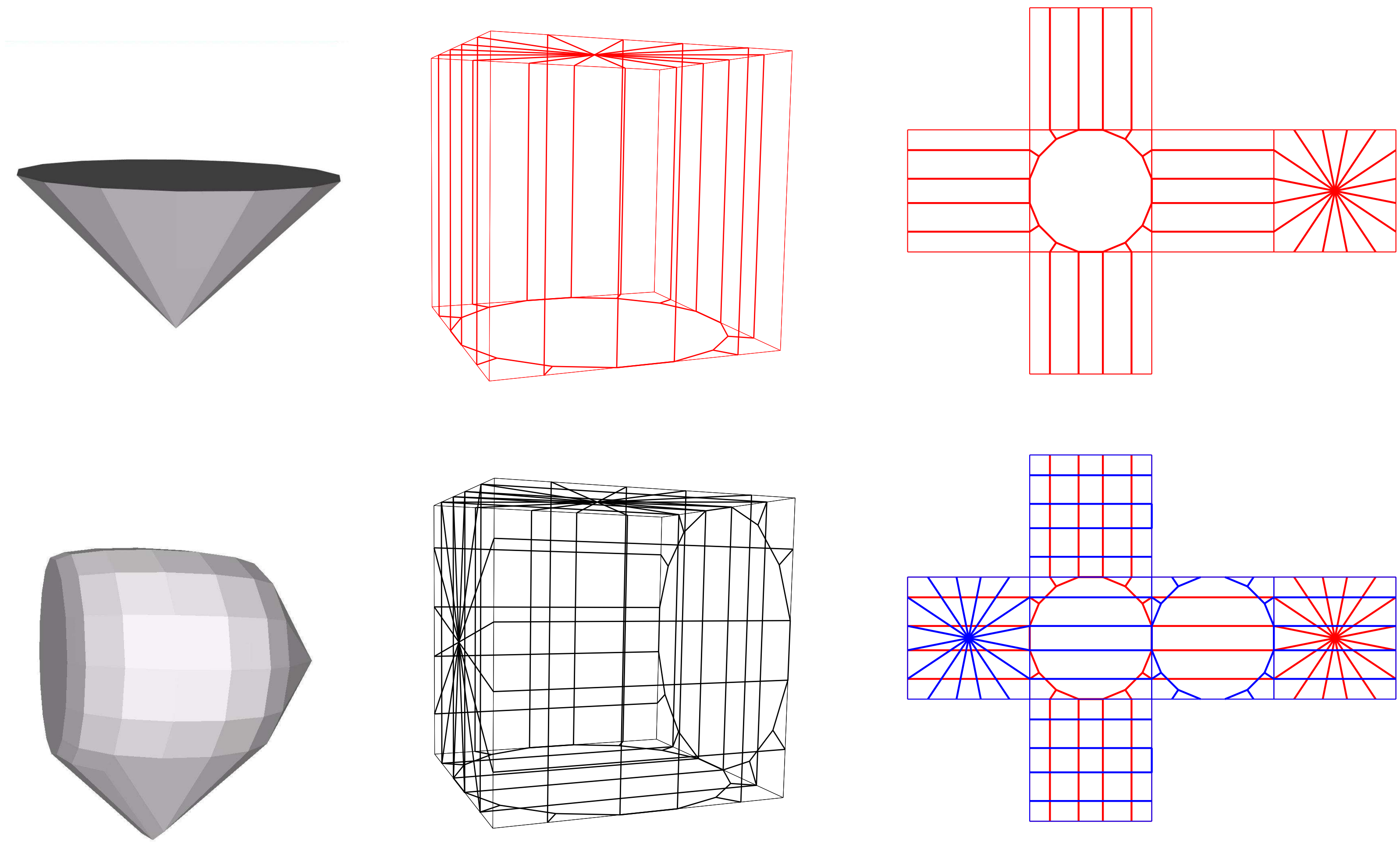
# Minkowski Sums of 3D Convex Polyhedra

The overlay of the Gaussian maps of two polytopes $P$ and $Q$ identifies all the pairs of features of $P$ and $Q$ respectively that have common supporting planes, as they occupy the same space on the unit sphere, thus, identifying all the pairwise features that contribute to the boundary of the Minkowski sum of $P$ and $Q$

- Each planar map in a CGM is a convex subdivision
- One algorithm that shows good results was implemented already
- It computes the Minkowski sum of *a set of polytopes*
- It runs it in $O(k \log(m + n))$ time

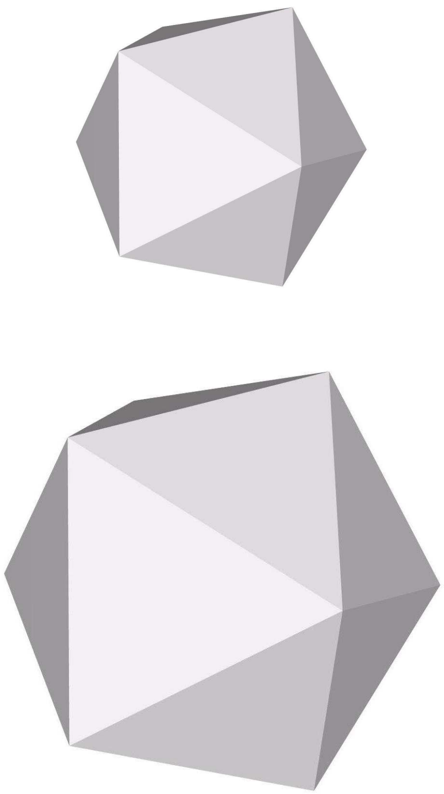
$m$, $n$, $k$ —number of facets in $P$, $Q$, $P \oplus Q$

# A Dioctagonal Pyramid and the Minkowski Sum of 2 Orthogonal Dioctagonal Pyramids

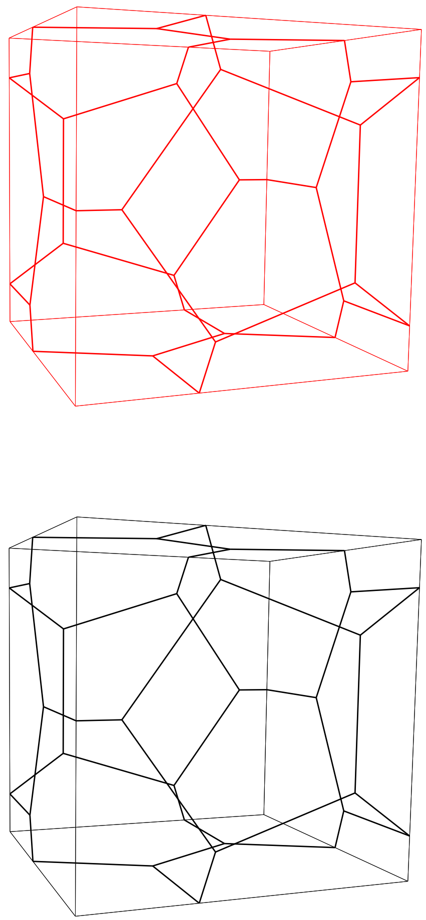


The primal | The CGM | The CGM unfolded

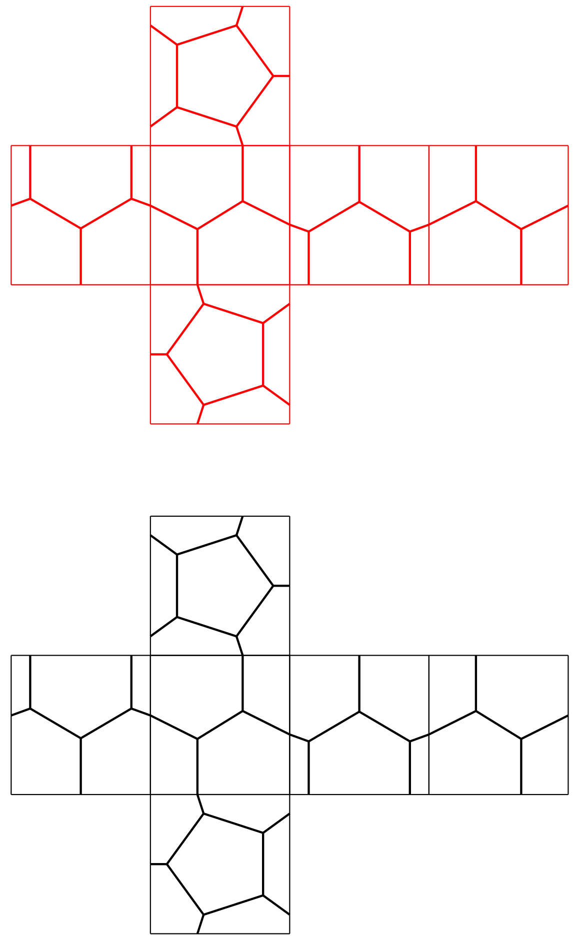
# An Icosahedron and the Minkowski Sum of 2 Identical Icosahedrons



The primal

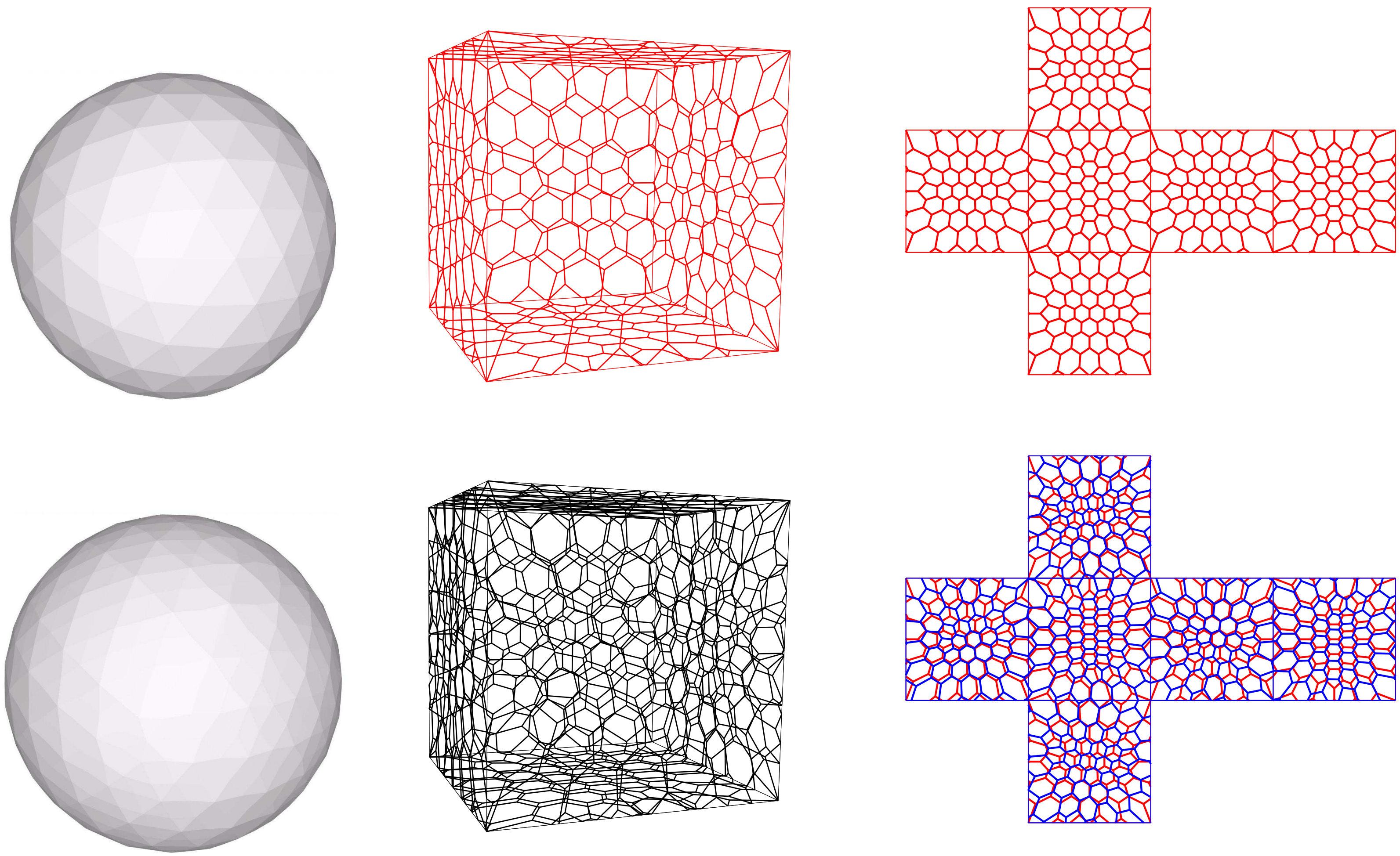


The CGM



The CGM unfolded

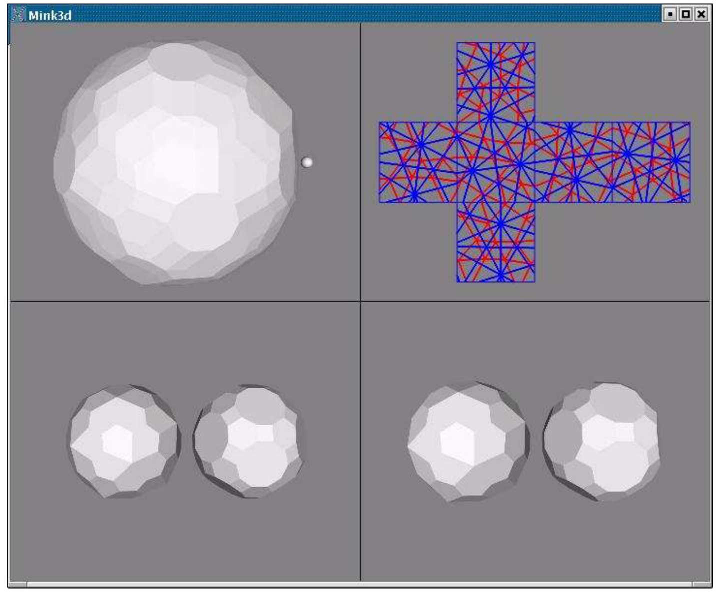
# A Geodesic Sphere and the Minkowski Sum of 2 Slightly Rotated Geodesic Spheres



The primal | The CGM | The CGM unfolded

# Minkowski Sums of Convex Polyhedra mink3d

An interactive program that simulates and demonstrates 3D collision detection



# Minkowski Sums of Convex Polyhedra Open Problems

- Exact complexity of $P \oplus Q$ with $n$, $m$ facets resp.
  - What is the exact upper bound?

    Conjecture: $\begin{cases} \frac{(m+1)(n+1)}{2} & mn \text{ is odd} \\ \frac{(m+1)(n+1)+1}{2} & mn \text{ is even} \end{cases}$
  - What is the exact lower bound when no degeneracies are present?
- Inverse problem given a convex polyhedron $M$
  - Is $M$ the valid *non-degenerate* Minkowski sum of a pair of convex polyhedra?
  - Compute the set of polyhedra pairs, such that the Minkowski sum of each pair is $M$

# Experimental Results

- The number of features of the six planar maps of the CGM of the squashed dioctagonal pyramid polytope

| **Planar map** | **V** | **HE** | **F** |
|---|---|---|---|
| 0, ($x = -1$) | 12 | 32 | 6 |
| 1, ($y = -1$) | 36 | 104 | 18 |
| 2, ($z = -1$) | 12 | 32 | 6 |
| 3, ($x = 1$) | 12 | 32 | 6 |
| 4, ($y = 1$) | 21 | 72 | 17 |
| 5, ($z = 1$) | 12 | 32 | 6 |
| **Total** | 105 | 304 | 59 |

# Experimental Results

- A data base of several hundreds models of convex polyhedra was created
- Complexity of the primal and dual representations

| **Object Name** | **Primal** | | | **Dual** | | |
|---|---|---|---|---|---|---|
| | **V** | **E** | **F** | **V** | **HE** | **F** |
| Tetrahedron | 4 | 6 | 4 | 38 | 94 | 21 |
| Octahedron | 6 | 12 | 6 | 24 | 48 | 12 |
| Icosahedron | 12 | 30 | 20 | 72 | 192 | 36 |
| Dioctagonal Pyramid | 17 | 32 | 17 | 105 | 304 | 59 |
| Pentagonal Hexecontahedron | 92 | 150 | 60 | 196 | 684 | 158 |
| Truncated Icosidodecahedron | 120 | 180 | 62 | 230 | 840 | 202 |
| Geodesic Sphere level 4 | 252 | 750 | 500 | 708 | 2124 | 366 |

# Experimental Results

- Time consumption of Minkowski-sum computation

| Object 1 | Object 2 | Minkowski Sum | | | | | | CH | CGMO |
|---|---|---|---|---|---|---|---|---|---|
| | | Primal | | | Dual | | | | |
| | | V | E | F | V | HE | F | | |
| Icosahedron | Icosahedron | 12 | 30 | 20 | 72 | 192 | 36 | 0.13 | 0.03 |
| Dioctagonal Pyramid | Orthogonal Dioctagonal Pyramid | 162 | 352 | 192 | 344 | 1136 | 236 | 0.53 | 0.28 |
| Pentagonal Hexeconta-hedron | Truncated Icosidodec-ahedron | 527 | 964 | 439 | 719 | 2744 | 665 | 16.76 | 1.21 |
| Geodesic Sphere level 4 | Rotated Geodesic Sphere level 4 | 814 | 1952 | 1146 | 1530 | 5060 | 1012 | 134.35 | 2.53 |

# 3D Collision Detection under Translation

$$P^u \cap Q^w \neq \emptyset \Leftrightarrow v - u \in M = P \oplus (-Q), \tag{1}$$

$$\delta(P^u, Q^w) = \min\{\|t\| \,|\, (v - u + t) \in M, t \in \mathbb{R}^3\}, \tag{2}$$

$$\pi_d(P^u, Q^w) = \inf\{\alpha \,|\, (v - u + \vec{d}\alpha) \notin M\}. \tag{3}$$

- Explicit computation of $M$ in a preprocessing phase
  - The combinatorial structure of $M$ is invariant to translations of $P$ and $Q$
- Implicit computation of $M$
  - Only portions of the boundary of $M$ necessary to answer the query are constructed

# 3D Collision Detection under Translation and Rotation

- Handle polytopes that undergo rigid motions
- Exploit spatial and temporal coherence between successive queries
  - Spatial - only the local neighborhood is considered to determine witnesses
  - Temporal - last answer is used as a hint in the next invocation
- Handle exact rotations in 3D
- Avoid Reconstruction of the CGM when the polytope is rotated
- Avoid construction of the entire Minkowski sum

# Main Research-Contributions

- Improved operations on planar arrangements
  - Map overlay in particular
- The CGM data structure & operations on it
- Minkowski sums in 3D
  - Exact & efficient computation
  - Understanding of certain theoretical issues
- Exact, efficient, & competitive collision detection and proximity queries of convex polyhedra that undergo rigid motions in 3D
- Efficient rational rotations in 3D

# The CGM Data-Structure