Robust Computational Geometry for Design, Manufacturing, and Robotics

Elisha Sacks, Purdue University

Victor Milenkovic, University of Miami

eps@cs.purdue.edu

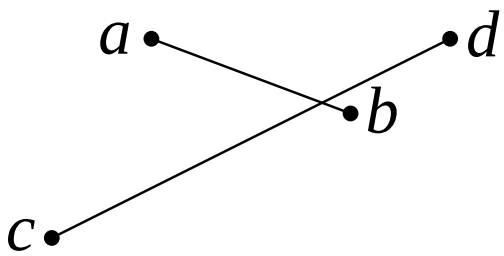
Problem

- Computational geometry develops algorithms for geometry and topology problems.
- The algorithms assume that arithmetic operations have infinite precision and constant time/space complexity.
- The *robustness problem* is how to implement the algorithms with computer arithmetic.
- Floating point arithmetic is fast, but inaccurate.
- Integer arithmetic is accurate, but slow.

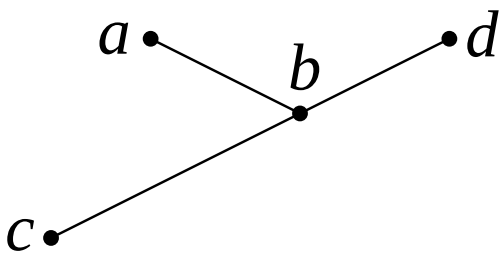
Robustness

Computational geometry algorithms branch on the signs of predicates. Unsafe predicates cause robustness problems.

- Predicate: $f(a) > 0$ means true; $f(a) < 0$ means false.
- Unsafe: $|f(a)|$ less than evaluation accuracy.
- Degenerate: $f(a) = 0$, always unsafe.



unsafe



degenerate

Inconsistency

- Unsafe predicates can lead to *inconsistent* outputs that are false for every input.
- Example: $a < b < c < a$ is inconsistent; sorting runs forever.
- Direct floating point implementation of real arithmetic causes inconsistency.
- Inconsistency causes algorithms to fail or output garbage.

Exact computational geometry

Implement predicates exactly using integer arithmetic.

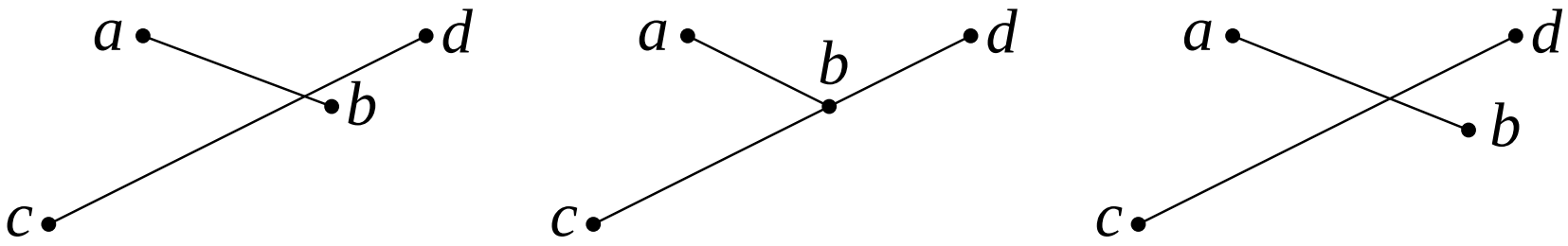
- Technical problems
  - Algebraic degree and bit complexity grow rapidly.
  - Slow complex software.
  - Degeneracy not handled.
- Conceptual problem
  - Engineering is approximate.
  - Numerical analysis and floating point work well.
  - Why should computational geometry be exact?

Approximate computational geometry

Implement predicates using floating point arithmetic.

- Inconsistency sensitivity.
  - Modify algorithm to ensure accuracy.
  - Advantage: speed and accuracy.
  - Problems: lack of generality, error analysis.
- Controlled linear perturbation.
  - Modify input to ensure accuracy.
  - Fast and accurate.
  - General purpose robustness module.
  - Degeneracy handled.

Perturbation strategy



unsafe degenerate perturbation

- Compute perturbed input that makes predicates safe.
- Error: distance between original and perturbed inputs.
- Advantages:
  - General.
  - Degeneracy handled transparently.

Perturbation algorithms

- Controlled perturbation.
  - Uniform perturbation.
  - Perturbation interval based on evaluation error.
  - Large errors.
  - No equality constraints or implicits.
- Controlled linear perturbation.
  - Optimal perturbation of linearized predicates.
  - Transfer results to original predicates.
  - Nearly minimal errors.
  - Equality constraints and implicits handled.

Controlled linear perturbation

- Given: predicate $f(\mathbf{x})$ with input $\mathbf{x} = \mathbf{a}$ and safety threshold ϵ.
- If f is safe, return the sign of $f(\mathbf{a})$.
- Pick perturbation direction $\mathbf{v}$.
- Select sign, s, of $\mathbf{v} \cdot \nabla f$.
- Compute $\mathbf{p} = \mathbf{a} + \delta\mathbf{v}$ that makes $f(\mathbf{p})$ safe.
  - $f(\mathbf{p}) \approx f(\mathbf{a}) + \delta(\mathbf{v} \cdot \nabla f)$
  - Solve $s[f(\mathbf{a}) + \delta(\mathbf{v} \cdot \nabla f)] \geq \epsilon$ for minimal δ.
  - This is a linear equation!
- Return s.

Sorting example

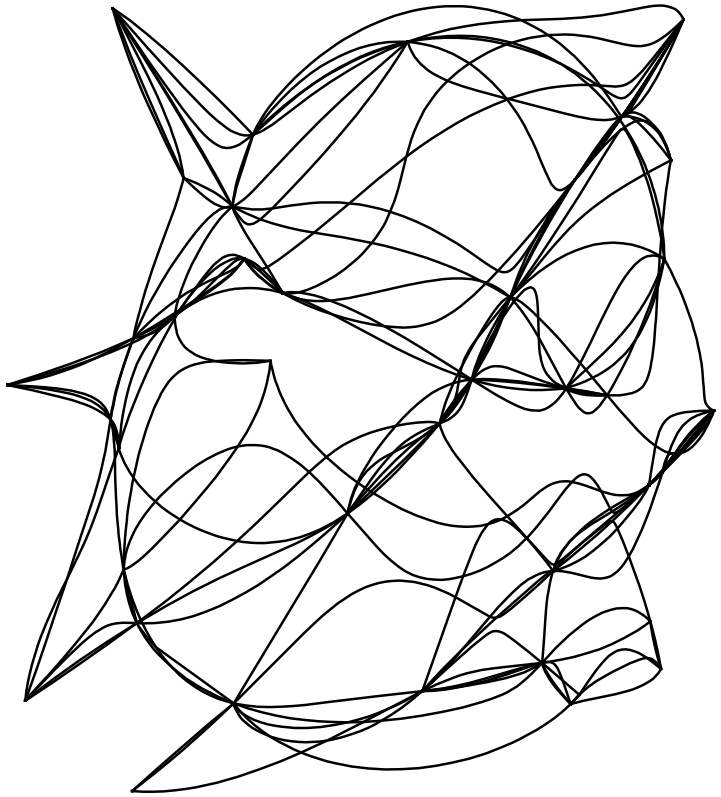
- Sort $\mathbf{x} = (x_1, x_2, x_3, x_4)$ with $\mathbf{a} = (0, 0, 0, 1)$.
- The $x_i < x_j$ predicate polynomial is $x_j - x_i$.
- The polynomials with $i, j < 4$ are unsafe.
- Their linearizations are $a_j + v_j - a_i - v_i$.
- Set $\mathbf{v} = (0.3, 0.8, 0.4, -1)$.
- The sign of $x_3 - x_1$ is 1 because $0.4 - 0.3 > 0$.
- The inequality is $0.1\delta \geq \epsilon$, so $\delta \geq 10\epsilon$
- The sign of $x_3 - x_2$ is -1 because $0.4 - 0.8 < 0$.
- The inequality is $0.4\delta \geq \epsilon$, so $\delta \geq 2.5\epsilon$
- The maximum, $\delta = 10\epsilon$, makes the input safe.

CLP versus controlled perturbation

- Comparison on convex hull and Delaunay triangulation.
- Controlled perturbation accuracy 3 digits.
- CLP accuracy 10–13 digits.
- Similar running times.
- Results transfer to other computational geometry algorithms.

CLP versus ECG

- Arrangement of 100 random degree-6 algebraic curves: 22 seconds with CLP; 220 seconds with ECG [Eigenwillig, 2008].
- Arrangement of 100 degenerate curves.



CLP versus ECG

- Arrangement contains 1330 vertices, including 43 clusters of nearly equal vertices with an average of 23 vertices per cluster and 55 vertices in the largest cluster.
- 1.5 seconds with CLP; estimated 30,000 seconds with ECG.
- Estimate based on measured root separation, ρ, and on published $\log^2 \rho$ running time.

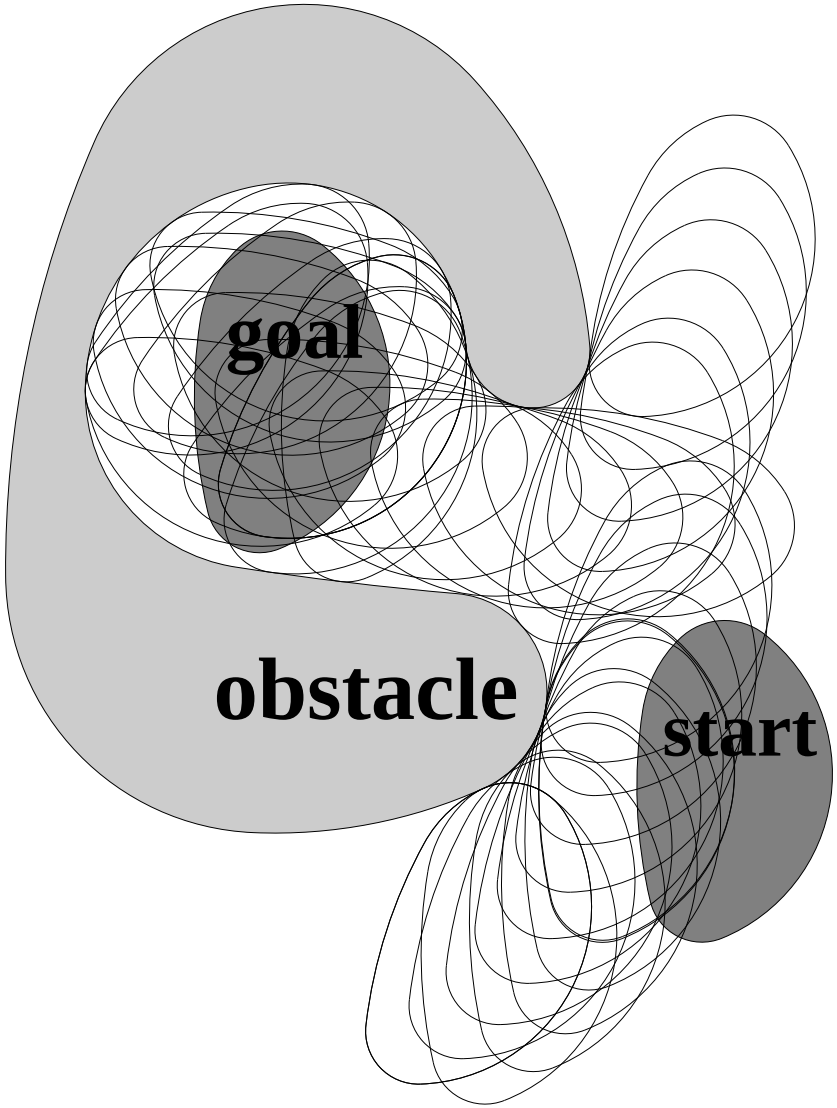
Research issues

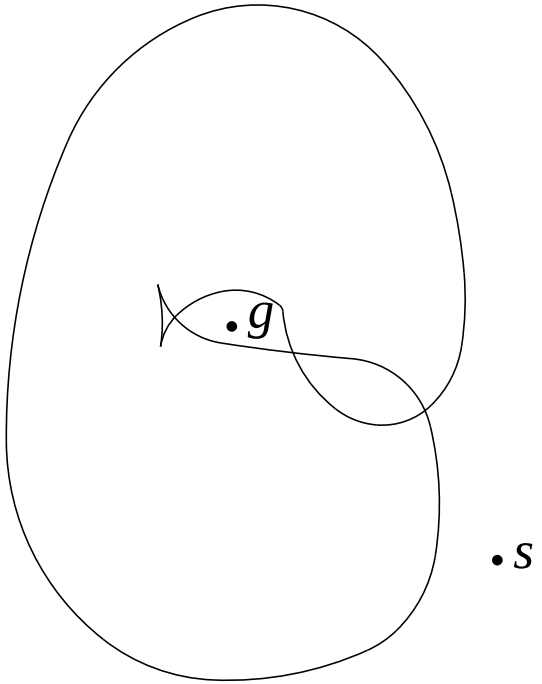
- Optimal versus random perturbation direction.
- Verification on original polynomials.
- Imposing equality constraints.
- Defining implicit variables.
- Handling singular predicates.
- Output simplification.

Applications

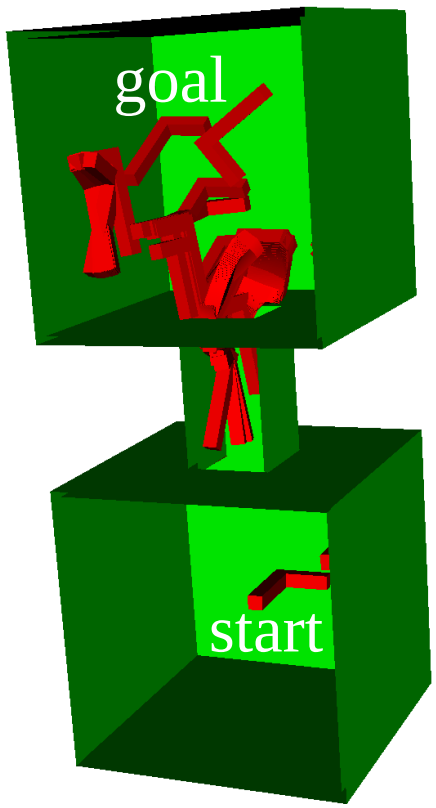
- Planar robot path planning (Milenkovic and Trac).
- Spatial robot path planning (Sacks and Milenkovic).
- Minkowski sums of polyhedra (Kyung).
- Mechanism design (Sacks).
- Part layout (Milenkovic).

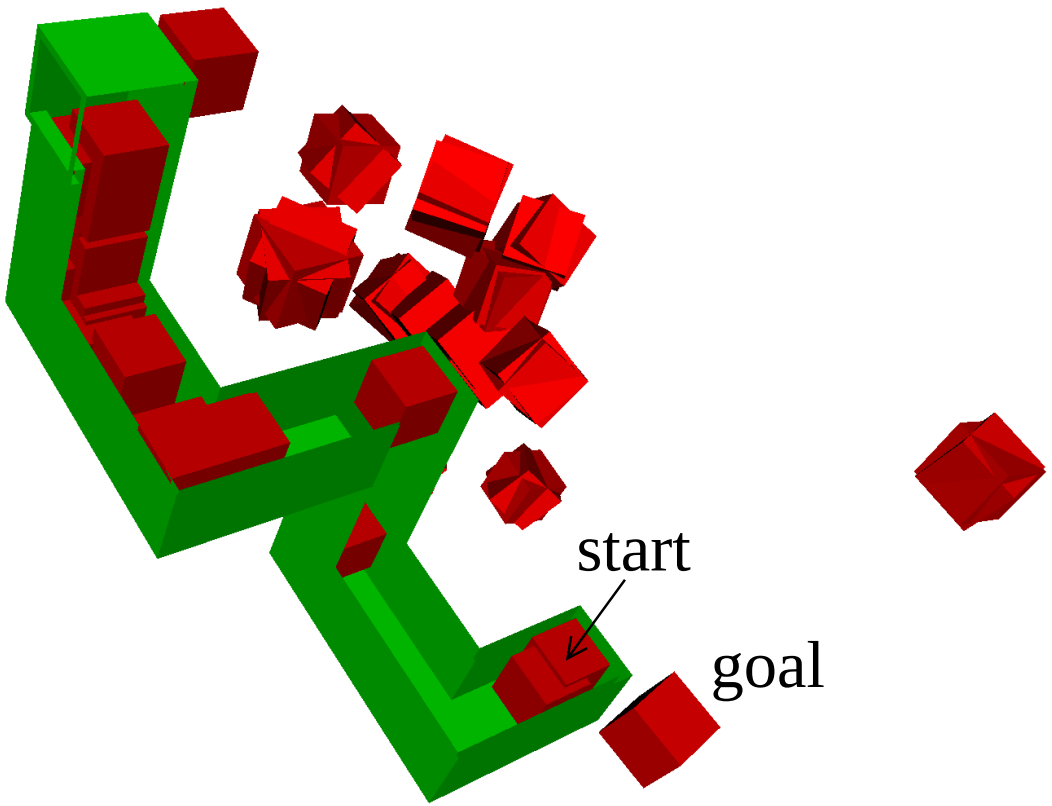
Planar robot path planning



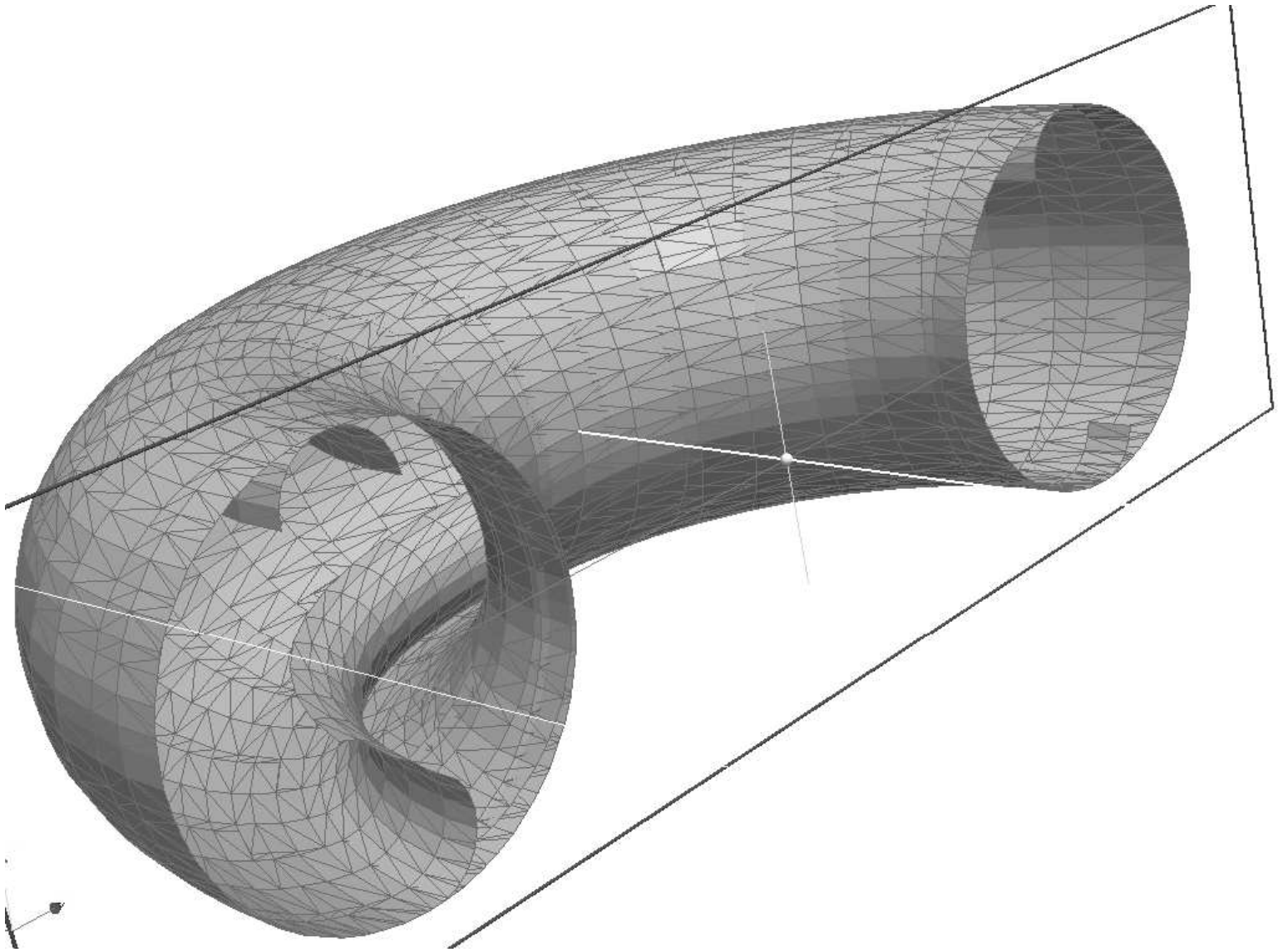


Spatial robot path planning

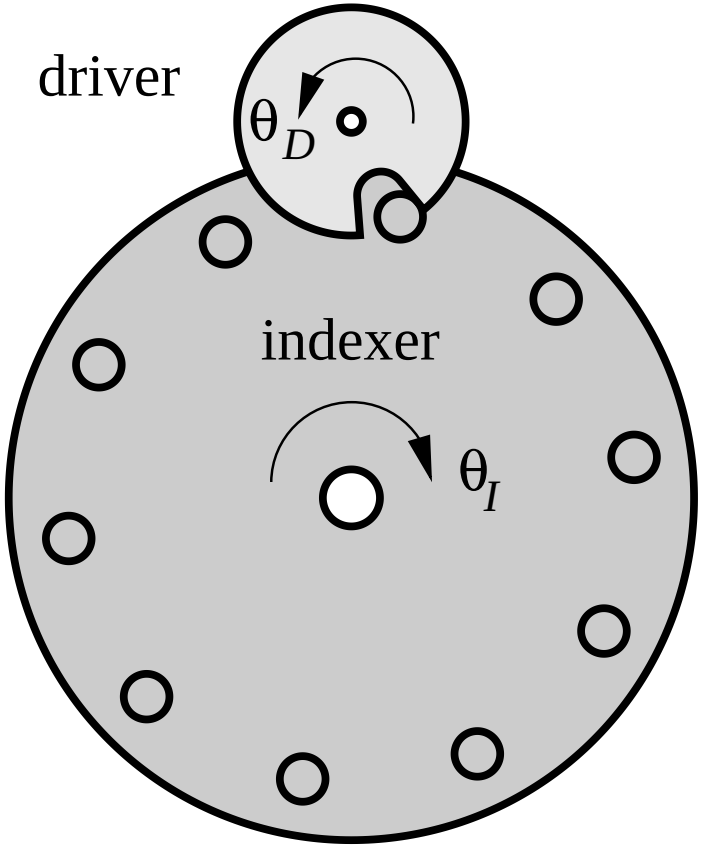


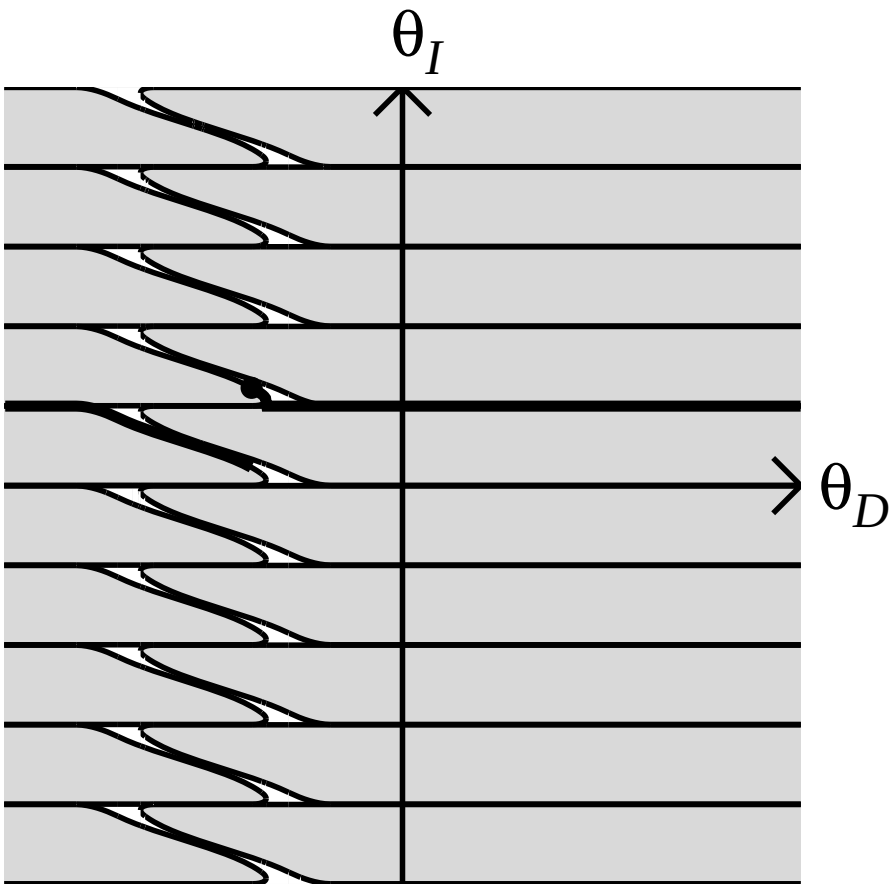


Minkowski sum of polyhedra



Mechanism design





Part layout