APPLIED aspects of COMPUTATIONAL GEOMETRY

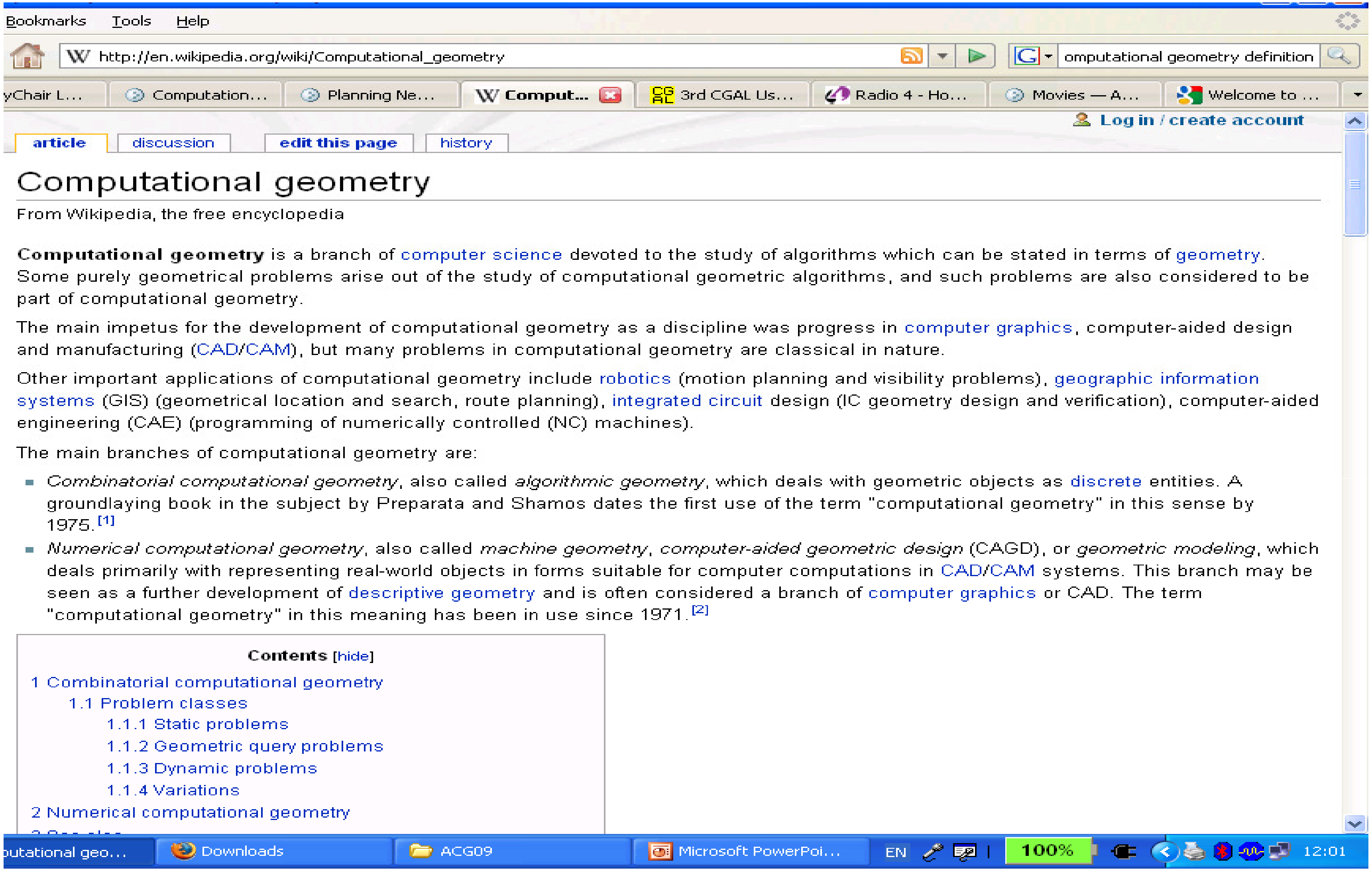
Introduction

Dan Halperin
School of Computer Science
Tel Aviv University

Lesson overview

- Background
- The main topics
- Course mechanics
- Additional topics

- Background
- The main topics
- Course mechanics
- Additional topics



Computational geometry

From Wikipedia, the free encyclopedia

Computational geometry is a branch of computer science devoted to the study of algorithms which can be stated in terms of geometry. Some purely geometrical problems arise out of the study of computational geometric algorithms, and such problems are also considered to be part of computational geometry.

The main impetus for the development of computational geometry as a discipline was progress in computer graphics, computer-aided design and manufacturing (CAD/CAM), but many problems in computational geometry are classical in nature.

Other important applications of computational geometry include robotics (motion planning and visibility problems), geographic information systems (GIS) (geometrical location and search, route planning), integrated circuit design (IC geometry design and verification), computer-aided engineering (CAE) (programming of numerically controlled (NC) machines).

The main branches of computational geometry are:

- *Combinatorial computational geometry*, also called *algorithmic geometry*, which deals with geometric objects as discrete entities. A groundlaying book in the subject by Preparata and Shamos dates the first use of the term "computational geometry" in this sense by 1975.[1]
- *Numerical computational geometry*, also called *machine geometry*, *computer-aided geometric design* (CAGD), or *geometric modeling*, which deals primarily with representing real-world objects in forms suitable for computer computations in CAD/CAM systems. This branch may be seen as a further development of descriptive geometry and is often considered a branch of computer graphics or CAD. The term "computational geometry" in this meaning has been in use since 1971.[2]

Contents [hide]

Computational geometry, standard assumptions

- computational model: the real RAM
- each basic operation on a small (constant-size) set of simple objects takes unit time
- general position
- these assumptions often do not hold in practice
- standard cs-theory asymptotic performance measures
- many times poor predictors of practical performance

Applied computational geometry

the goal:

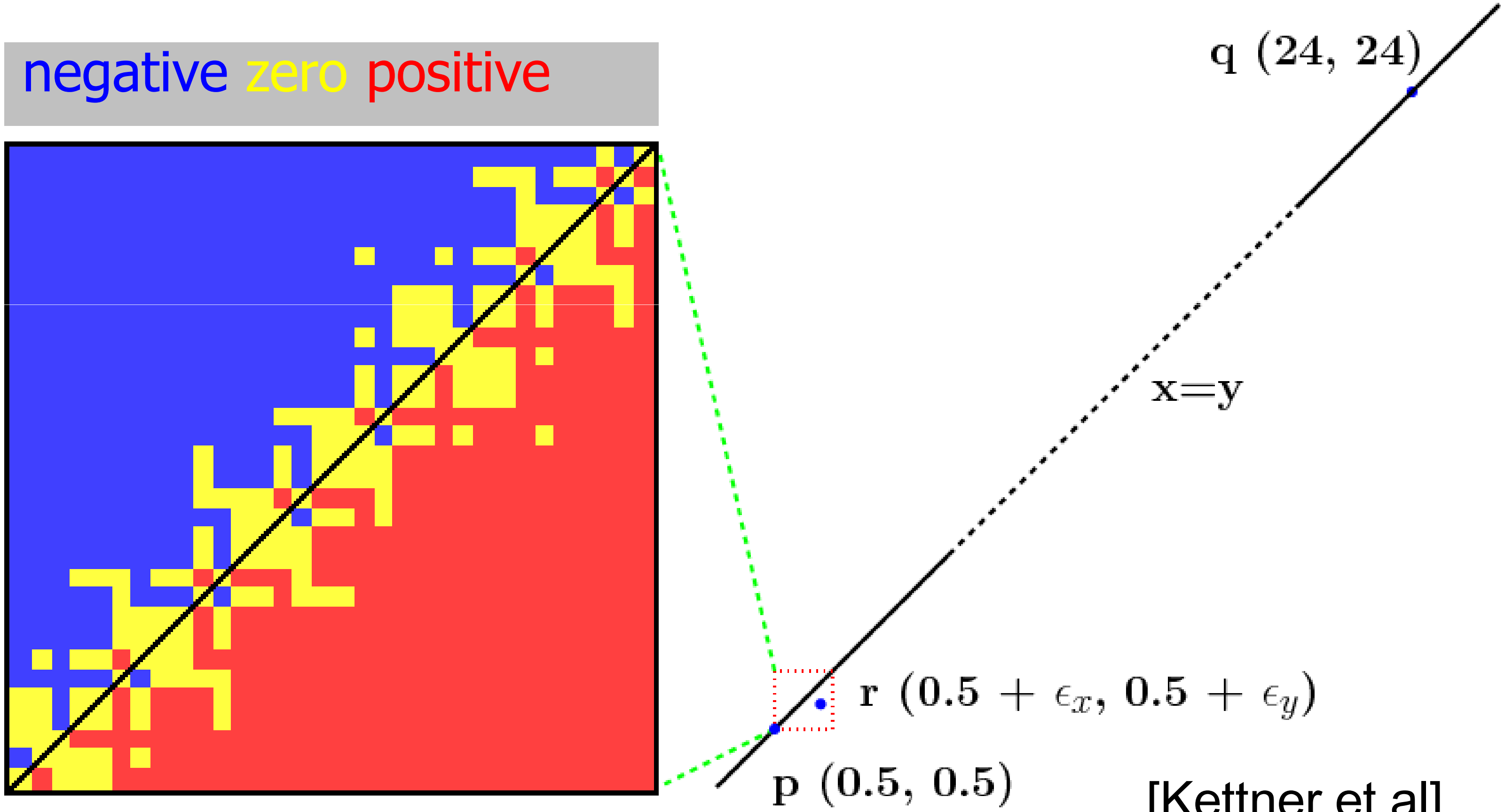
(re)design and implement geometric algorithms and data structures that are at once certified and efficient in practice

What’s the problem?

Q: Given two lines l1 and l2 in the plane, does the line l1 pass through the intersection point l1 ∩ l2?

What’s the problem? cont’d

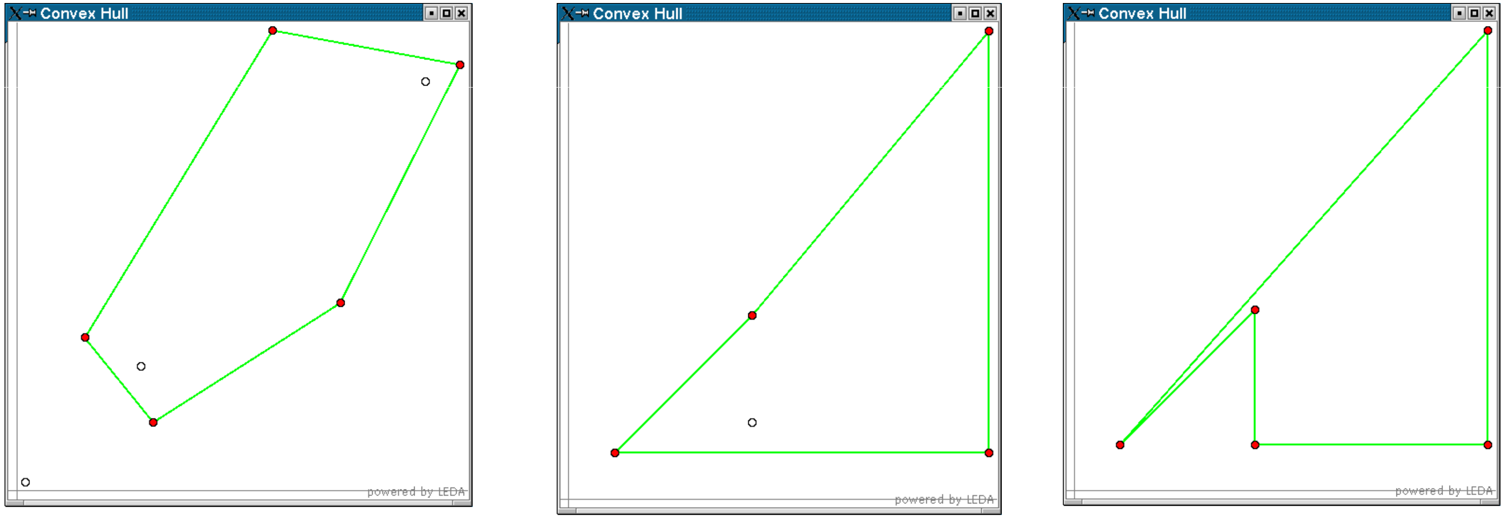
orientation(p,q,r) = sign((px-rx)(qy-ry)-(py-ry)(qx-rx))



[Kettner et al]

What’s the problem? cont’d

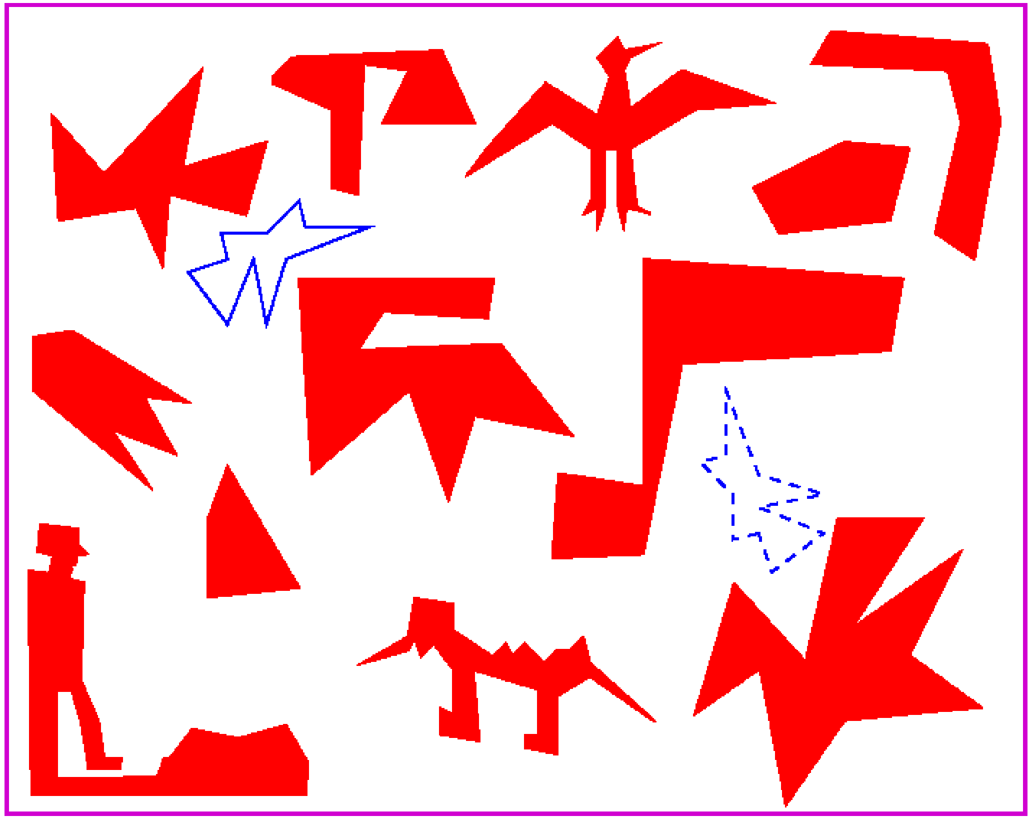
CG algorithms strongly couple numerical and combinatorial/topological data



[Kettner et al]

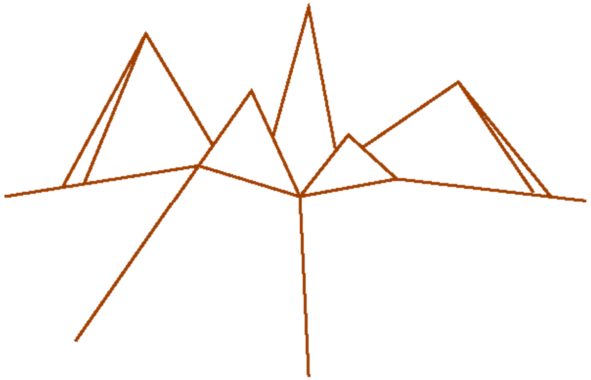
- Background
- The main topics
- Course mechanics
- Additional topics

Problem 1: Motion planning



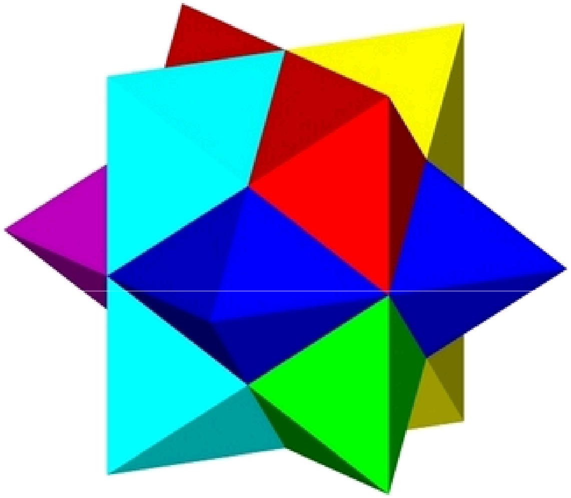
decide whether a collision-free motion for the moving object from start to goal exists, and if so plan the motion

Problem 2:
Aspect graph of a terrain



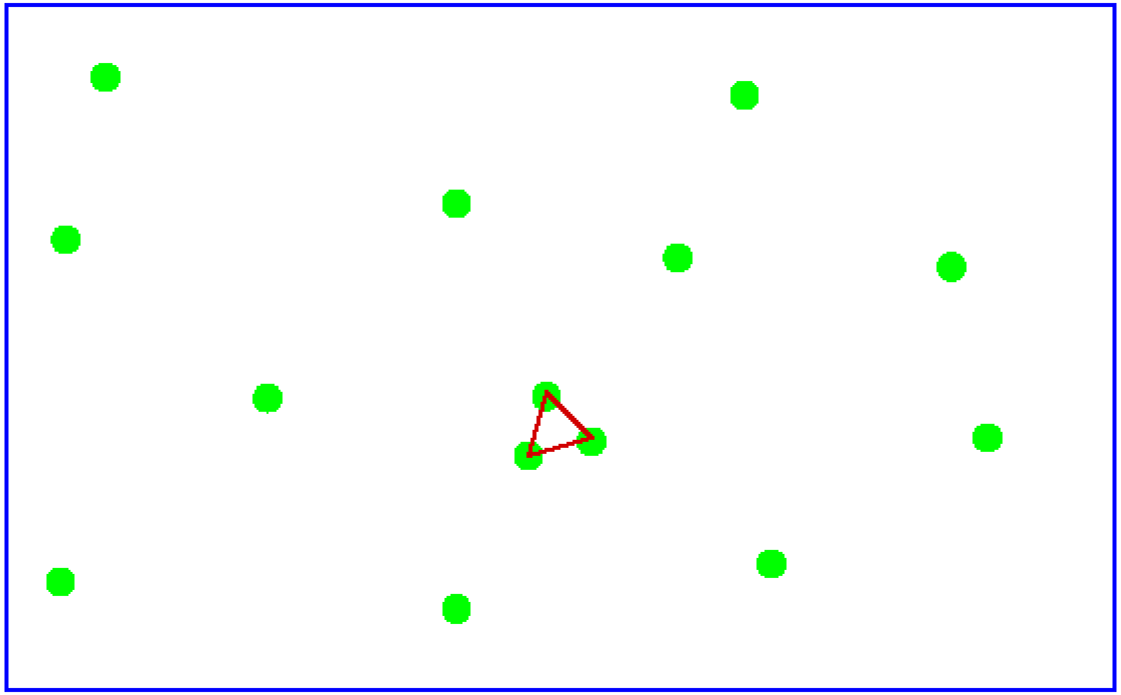
design a compact representation of all the different 2D images of a polyhedral terrain, so that the view in a given query direction can be efficiently retrieved

Problem 3:
Is the 3D object interlocked?



decide whether an assembled object is interlocked, namely cannot be taken apart with two hands

Problem 4: Minimum area triangle



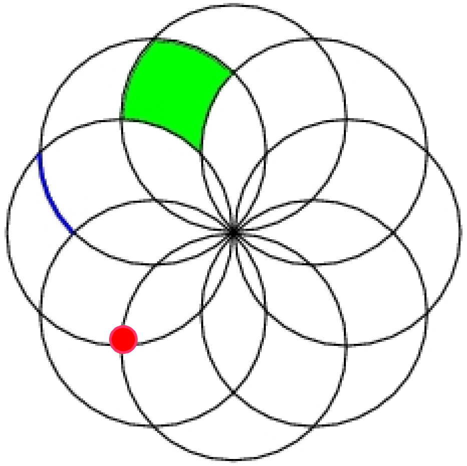
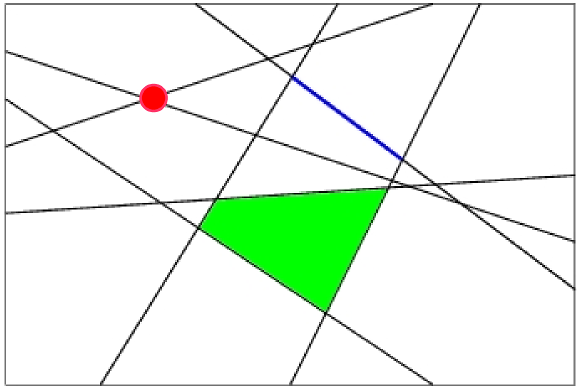
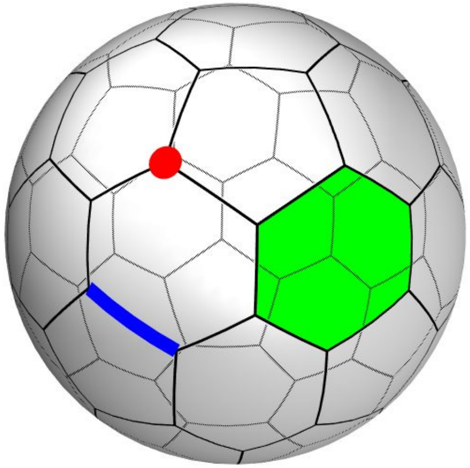
find the three of the given points that define the minimum area triangle

Q: What is the connection between Problems 1,2,3 and 4?

A: The best solution known to each of them was obtained with arrangements

Topic I: Arrangements of curves and surfaces

given a collection of curves on a surface, the **arrangement** is the partition of the surface into **vertices**, **edges** and **faces** induced by the curves

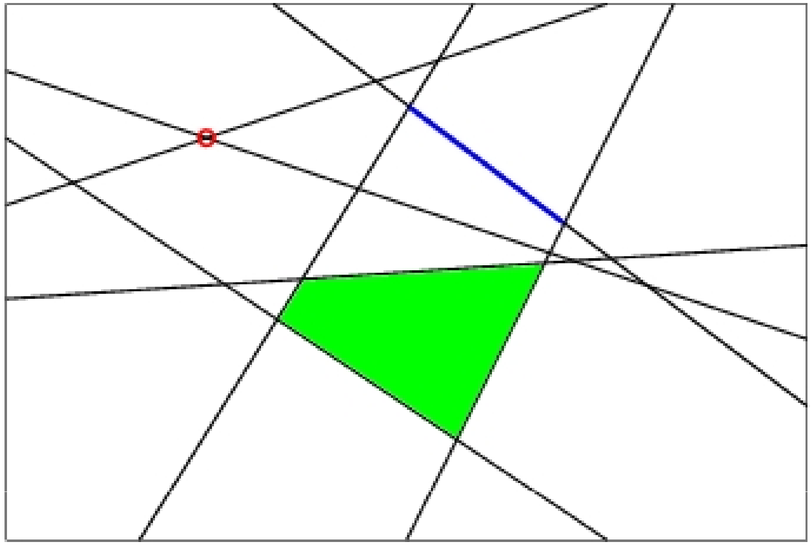
What are arrangements?

- an arrangement of a set S of geometric objects is the subdivision of space where the objects reside induced by S
- possibly non-linear objects (circles), bounded objects (segments), higher dimensions (planes, simplices)
- numerous applications in robotics, molecular biology,vision, graphics, CAD/CAM, statistics, GIS
- have been studied for decades - Matoušek (2002) cites Steiner,1826; nowadays studied in combinatorial and computational geometry

Solving it with arrangements

- transforming to arrangements
- combinatorial analysis
- design of data structures / algorithms
- implementation

Arrangements of lines: combinatorics



the complexity of an arrangement is the overall number of cells of all dimensions comprising the arrangement

for planar arrangements we count vertices, edges, and faces

Q: what is the complexity of an arrangement of n lines?

Basic theorem of arrangement complexity

- the maximum combinatorial complexity of an arrangement of n well-behaved curves in the plane is O(n^2); there are such arrangements whose complexity is Ω(n^2)

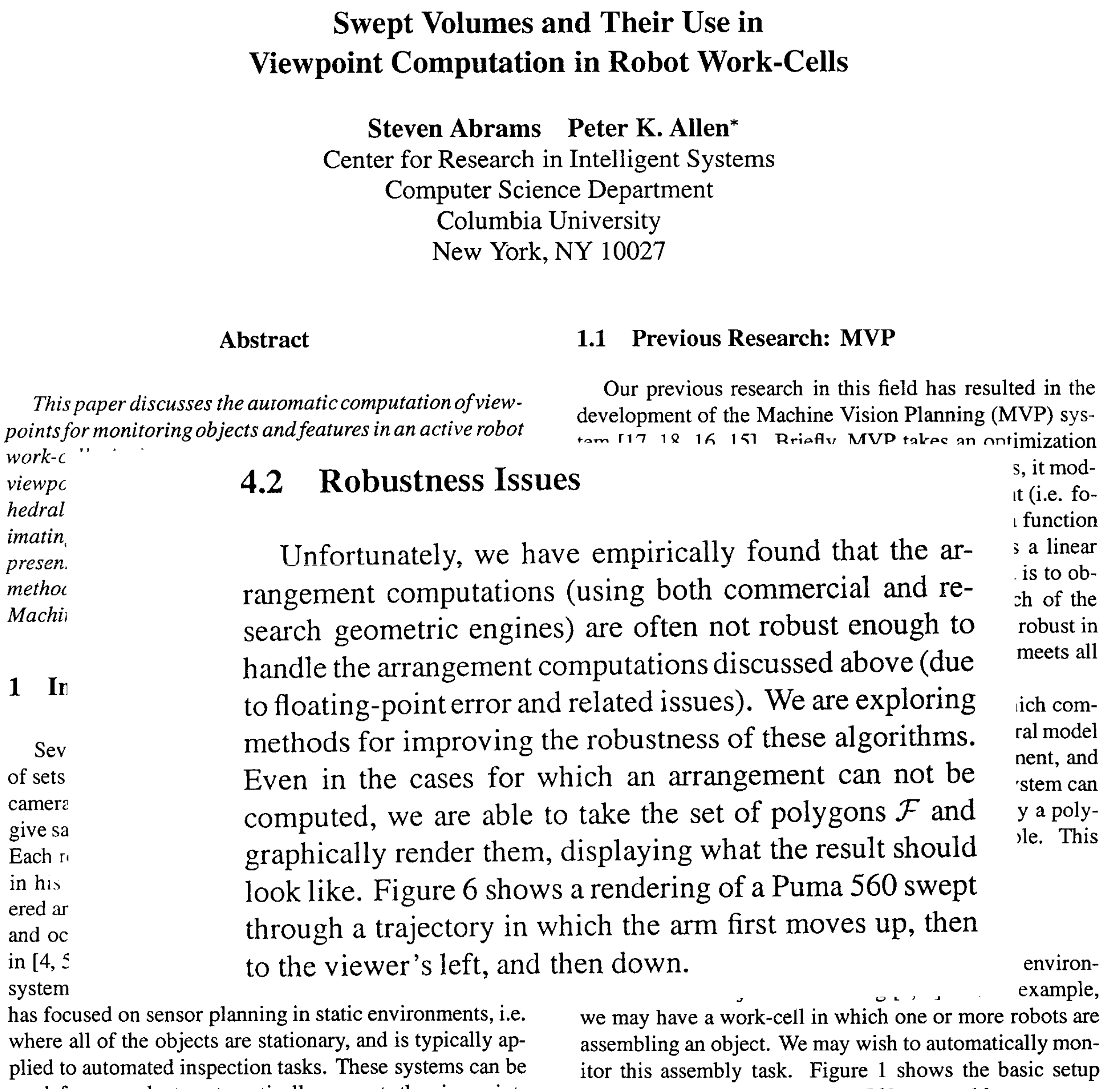
- more generally

  the maximum combinatorial complexity of an arrangement of n well-behaved (hyper)surfaces in R^d is $O(n^d)$; there are such arrangements whose complexity is $\Omega(n^d)$

Arrangements in the course

- arrgs underlie each of the main topics and some of the `additional topics’
- the practice of 2D arrangements, progress and experience; CGAL’s arrgs package
- envelopes of surfaces (~2.5D)

- constructing 3D arrangements
- coping with higher-dimensional arrangements (with applications of 4D and 5D arrgs)



Swept Volumes and Their Use in Viewpoint Computation in Robot Work-Cells

Steven Abrams Peter K. Allen*
Center for Research in Intelligent Systems
Computer Science Department
Columbia University
New York, NY 10027

Abstract

This paper discusses the automatic computation of viewpoints for monitoring objects and features in an active robot work-c
viewpc
hedral
imatin
presen
methoc
Machii

1 In

Sev
of sets
camera
give sa
Each r
in his
ered ar
and oc
in [4, 5
system
has focused on sensor planning in static environments, i.e. where all of the objects are stationary, and is typically applied to automated inspection tasks. These systems can be

1.1 Previous Research: MVP

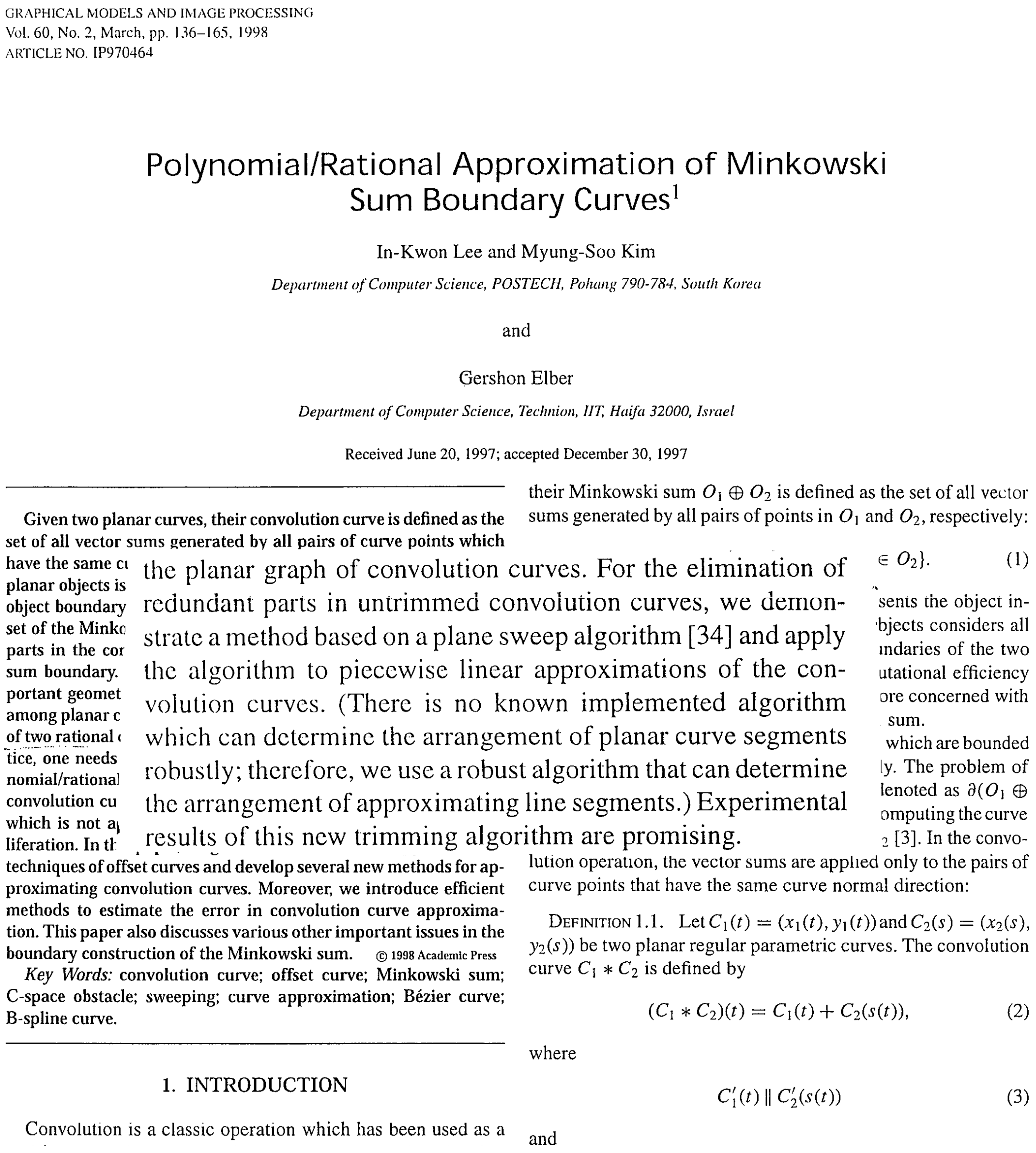
Our previous research in this field has resulted in the development of the Machine Vision Planning (MVP) system [17, 18, 16, 15]. Briefly, MVP takes an optimization
s, it mod-
it (i.e. fo-
function
a linear
is to ob-
ch of the
robust in
meets all

ich com-
ral model
nent, and
stem can
y a poly-
le. This

environ-
example,
we may have a work-cell in which one or more robots are assembling an object. We may wish to automatically monitor this assembly task. Figure 1 shows the basic setup

4.2 Robustness Issues

Unfortunately, we have empirically found that the arrangement computations (using both commercial and research geometric engines) are often not robust enough to handle the arrangement computations discussed above (due to floating-point error and related issues). We are exploring methods for improving the robustness of these algorithms. Even in the cases for which an arrangement can not be computed, we are able to take the set of polygons $\mathcal{F}$ and graphically render them, displaying what the result should look like. Figure 6 shows a rendering of a Puma 560 swept through a trajectory in which the arm first moves up, then to the viewer's left, and then down.



GRAPHICAL MODELS AND IMAGE PROCESSING
Vol. 60, No. 2, March, pp. 136–165, 1998
ARTICLE NO. IP970464

Polynomial/Rational Approximation of Minkowski Sum Boundary Curves[1]

In-Kwon Lee and Myung-Soo Kim

Department of Computer Science, POSTECH, Pohang 790-784, South Korea

and

Gershon Elber

Department of Computer Science, Technion, IIT, Haifa 32000, Israel

Received June 20, 1997; accepted December 30, 1997

Given two planar curves, their convolution curve is defined as the set of all vector sums generated by all pairs of curve points which
have the same c
planar objects is
object boundary
set of the Minko
parts in the cor
sum boundary.
portant geomet
among planar c
of two rational
tice, one needs
nomial/rational
convolution cu
which is not a
liferation. In th
techniques of offset curves and develop several new methods for approximating convolution curves. Moreover, we introduce efficient methods to estimate the error in convolution curve approximation. This paper also discusses various other important issues in the boundary construction of the Minkowski sum. © 1998 Academic Press

Key Words: convolution curve; offset curve; Minkowski sum; C-space obstacle; sweeping; curve approximation; Bézier curve; B-spline curve.

the planar graph of convolution curves. For the elimination of redundant parts in untrimmed convolution curves, we demonstrate a method based on a plane sweep algorithm [34] and apply the algorithm to piecewise linear approximations of the convolution curves. (There is no known implemented algorithm which can determine the arrangement of planar curve segments robustly; therefore, we use a robust algorithm that can determine the arrangement of approximating line segments.) Experimental results of this new trimming algorithm are promising.

1. INTRODUCTION

Convolution is a classic operation which has been used as a

their Minkowski sum $O_1 \oplus O_2$ is defined as the set of all vector sums generated by all pairs of points in O_1 and O_2, respectively:

$$\in O_2\}. \qquad (1)$$

sents the object in-
bjects considers all
ndaries of the two
utational efficiency
ore concerned with
sum.
which are bounded
y. The problem of
lenoted as $\partial(O_1 \oplus$
omputing the curve
2 [3]. In the convolution operation, the vector sums are applied only to the pairs of curve points that have the same curve normal direction:

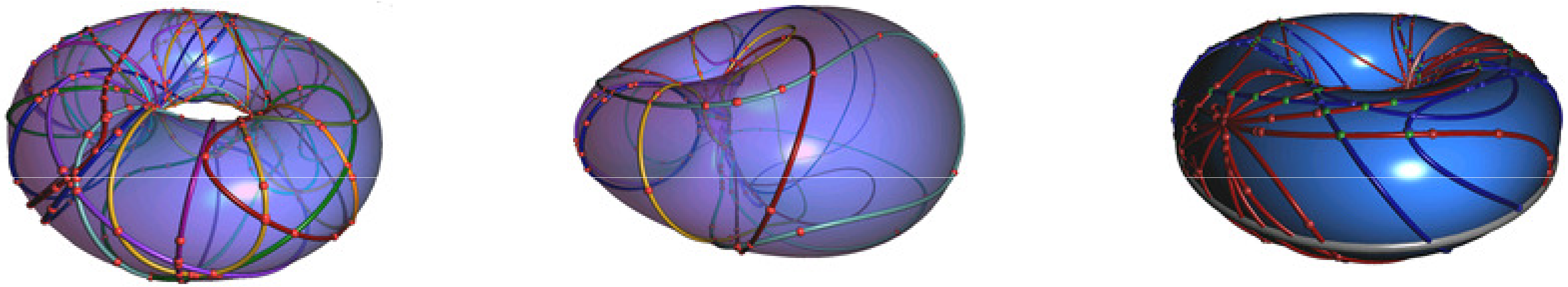
DEFINITION 1.1. Let $C_1(t) = (x_1(t), y_1(t))$ and $C_2(s) = (x_2(s), y_2(s))$ be two planar regular parametric curves. The convolution curve $C_1 * C_2$ is defined by

$$(C_1 * C_2)(t) = C_1(t) + C_2(s(t)), \qquad (2)$$

where

$$C_1'(t) \parallel C_2'(s(t)) \qquad (3)$$

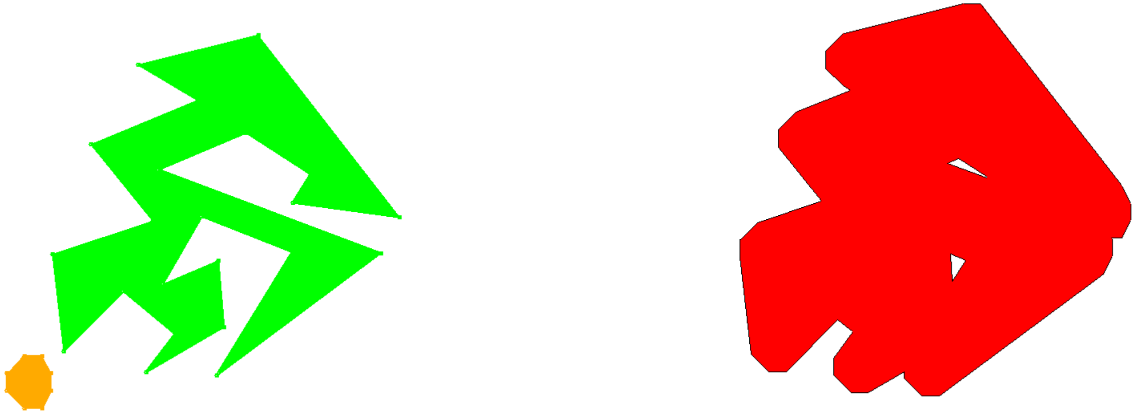
and



[Berberich et al]

Topic II: Minkowski sums

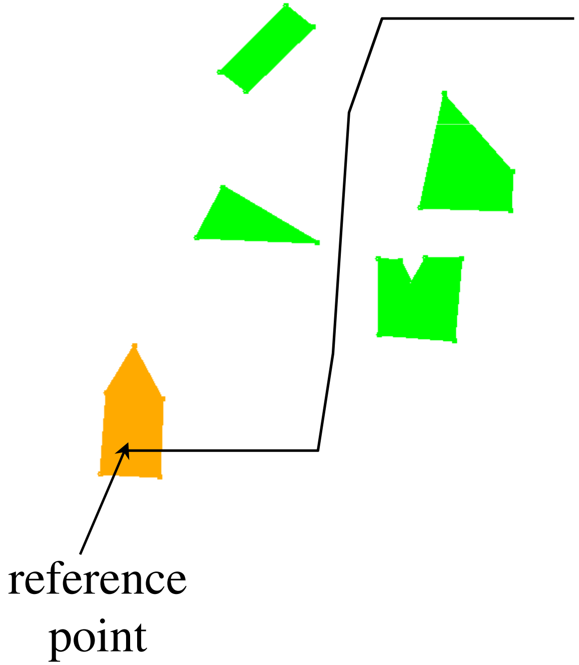
- Given two sets *P* and *Q*, *their Minkowski sum* $P \oplus Q = \{p+q \mid p \in P, q \in Q\}$
- When *P* and *Q* are polygonal sets, their Minkowski sum is a polygonal planar map

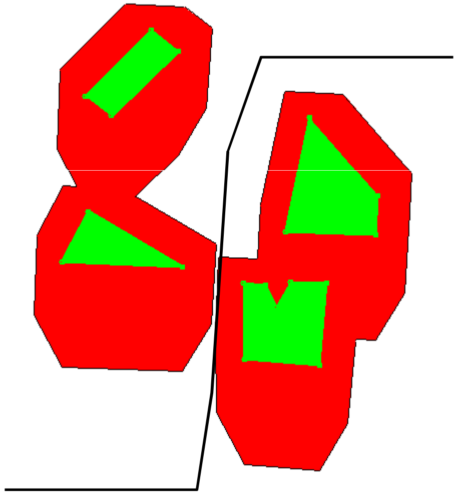


Typical usage: collision detection

Q - a polygonal object that moves by translation

P - a set of polygonal obstacles

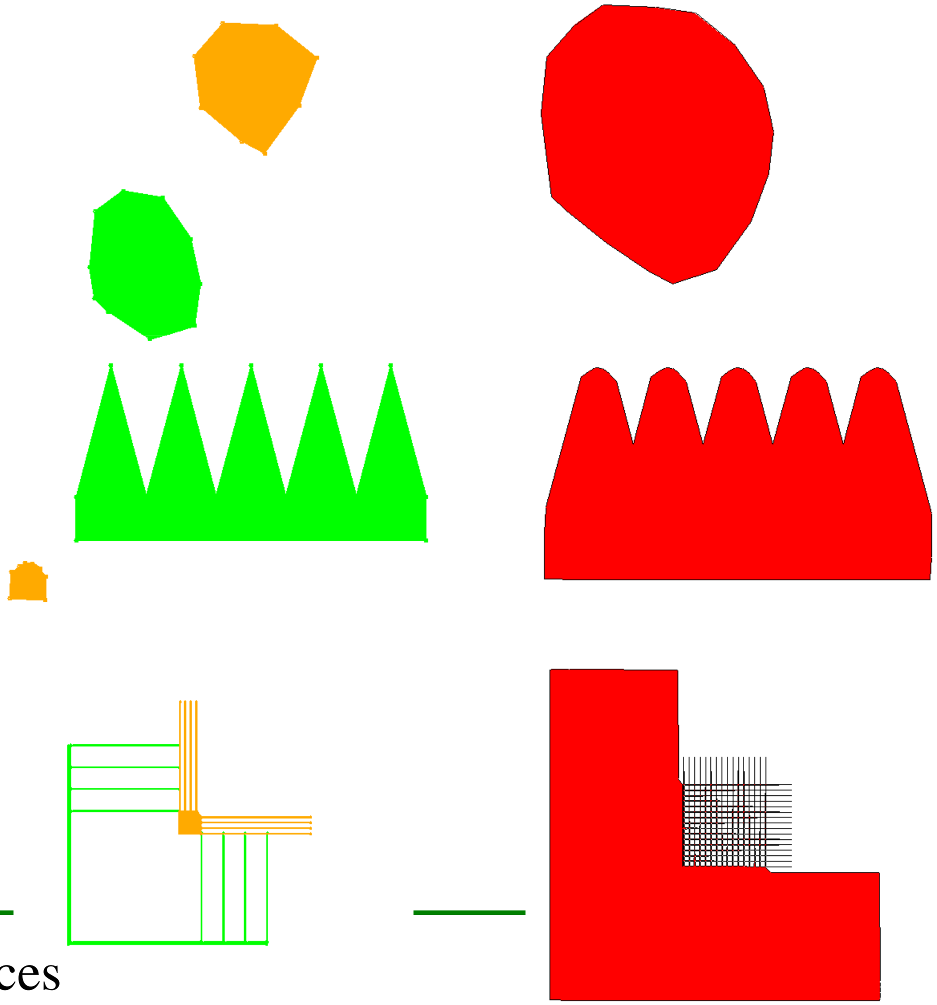




Claim: When translating, Q intersects P iff ref(Q) is inside P ⊕ -Q

Fundamental complexity bounds

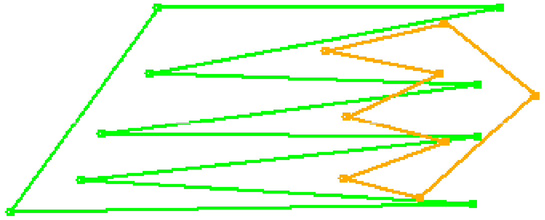
input	sum complexity
P is convex Q is convex	$\Theta(m+n)$
P is convex Q is general	$\Theta(m\ n)$ [KLPS]
P is general Q is general	$\Theta(m^2\ n^2)$



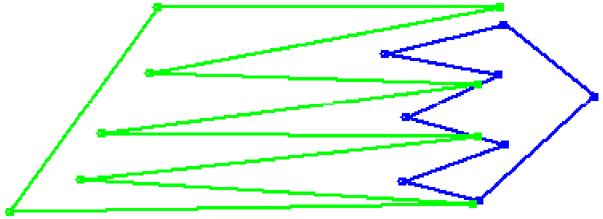
P with m vertices, Q with n vertices

Applications of Minkowski sums: Minimum distance separation

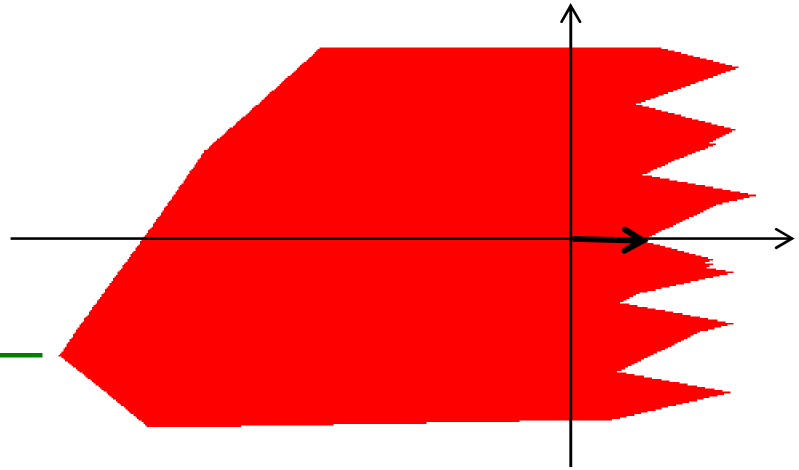
Translate the small polygon *P* such that it will not penetrate *Q*



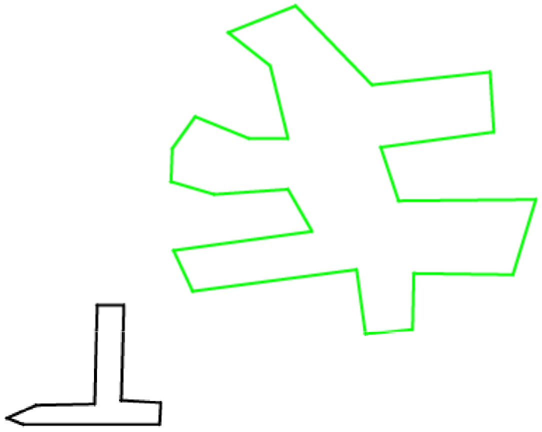
Separated polygons

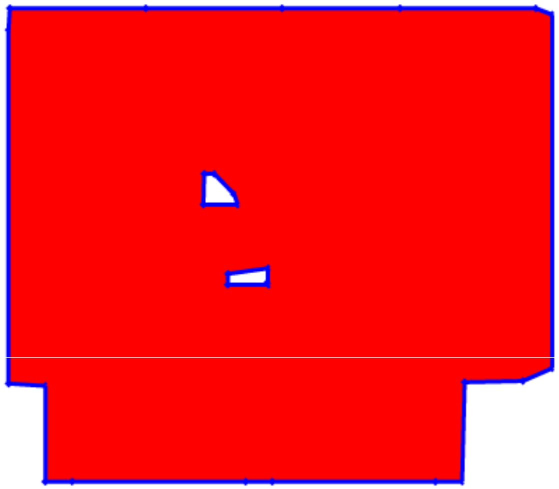


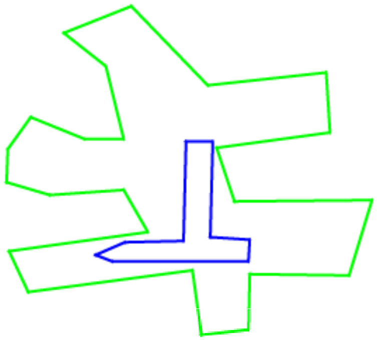
Find the closest point to the origin that is outside $Q \oplus -P$



Applications of Mink sums, cont'd: Polygon containment



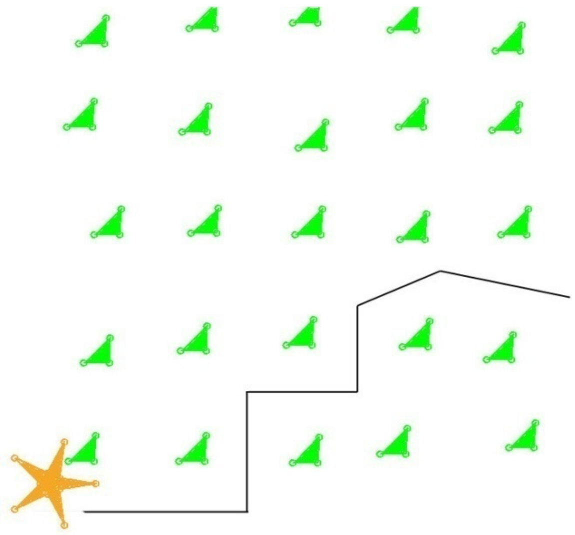




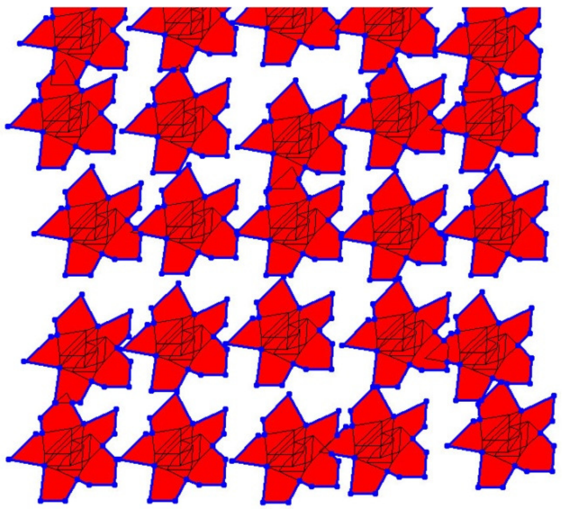
Can a polygon *P* be place inside another polygon *Q*?

Compute **(*B* \ *Q*) ⊕ -*P***:
(*B* is a bounding box of *Q*)
P can be placed inside *Q* when the reference point is placed in one of the holes

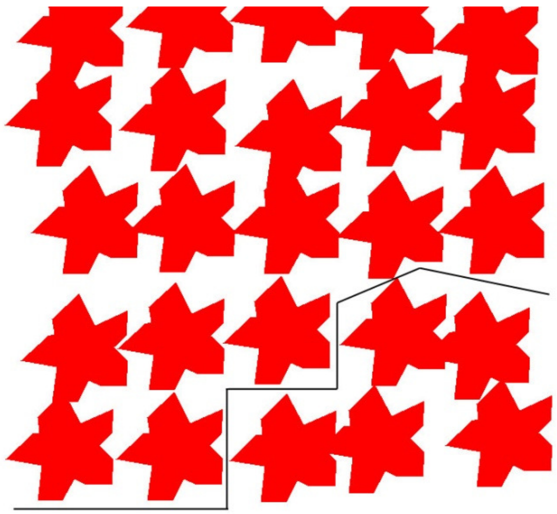
Applications of Mink sums, cont’d: Robot motion planing



robot, obstacles and computed path



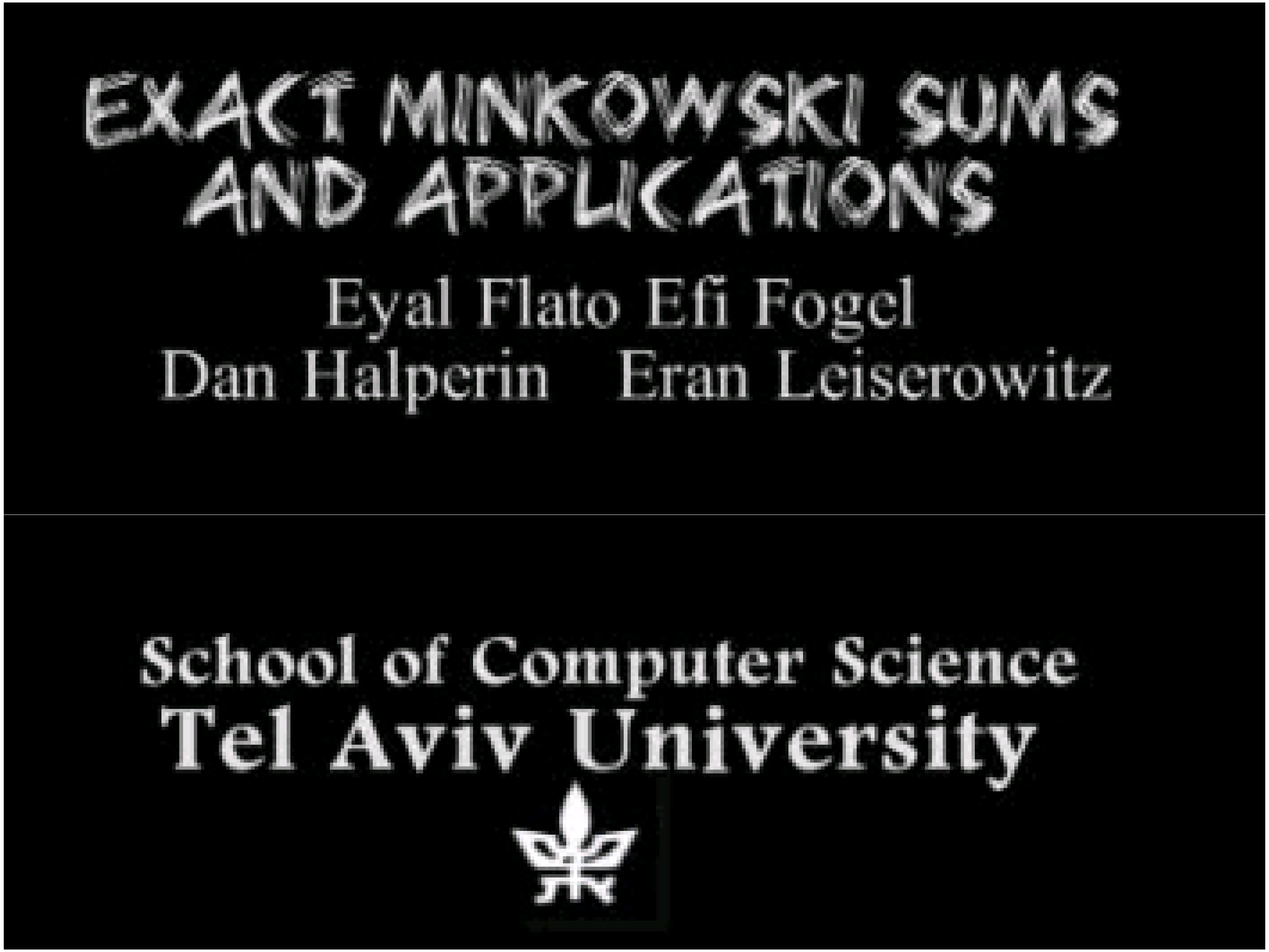
computed planar map



Minkowski sum of the robot with obstacles

Minkowski sums of convex polygons

- properties
- complexity
- algorithm (overlaying 1D arrangements)



EXACT MINKOWSKI SUMS AND APPLICATIONS

Eyal Flato Efi Fogel
Dan Halperin Eran Leiscrowitz

School of Computer Science
Tel Aviv University

Minkowski sums in the course

- the general polygonal case: theory and practice
- offset polygons
- summing 3-polytopes

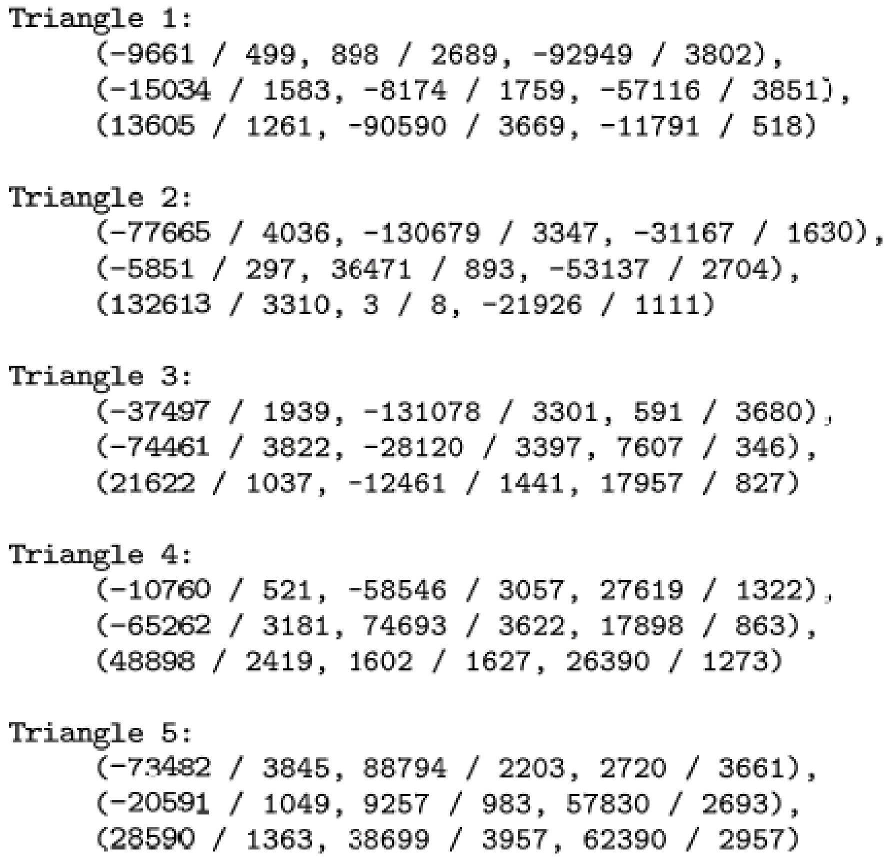
- the general polyhedral case: current state and challenges

Minkowski sums under rotation, video [Jyh Ming Lien]

Topic III: Geometric rounding

- EGC: the exact geometric computing paradigm
- EGC vs. fixed-precision approximation
- number types in geometric computing
- who needs rounding
- what is difficult about rounding

Complexity of numbers, input coordinates



```
Triangle 1:
     (-9661 / 499, 898 / 2689, -92949 / 3802),
     (-15034 / 1583, -8174 / 1759, -57116 / 3851),
     (13605 / 1261, -90590 / 3669, -11791 / 518)

Triangle 2:
     (-77665 / 4036, -130679 / 3347, -31167 / 1630),
     (-5851 / 297, 36471 / 893, -53137 / 2704),
     (132613 / 3310, 3 / 8, -21926 / 1111)

Triangle 3:
     (-37497 / 1939, -131078 / 3301, 591 / 3680),
     (-74461 / 3822, -28120 / 3397, 7607 / 346),
     (21622 / 1037, -12461 / 1441, 17957 / 827)

Triangle 4:
     (-10760 / 521, -58546 / 3057, 27619 / 1322),
     (-65262 / 3181, 74693 / 3622, 17898 / 863),
     (48898 / 2419, 1602 / 1627, 26390 / 1273)

Triangle 5:
     (-73482 / 3845, 88794 / 2203, 2720 / 3661),
     (-20591 / 1049, 9257 / 983, 57830 / 2693),
     (28590 / 1363, 38699 / 3957, 62390 / 2957)
```

Complexity of numbers, computed coordinates

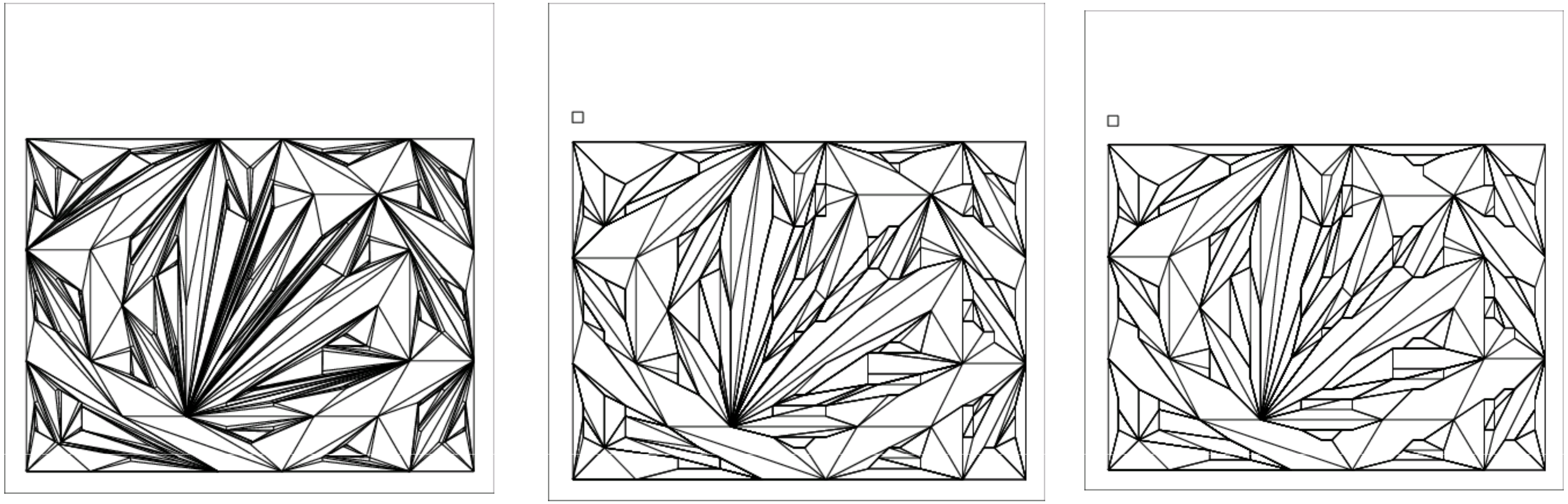
A normalized coordinate of the worst feature of the partial decomposition — 237 digits long:

```
PD feature = 49799838826104887192775516219046994702
461828025059123646217485873346921099238939609590257
26989674024022169299702332971 / 5027790709859107937
563103744532644005619919434042984323896243977724409
28440717068821348688514967315807043013459806716
```

A normalized coordinate of the worst feature of the full decomposition — 559 digits long:

```
FD feature = 23279315243924676155798958688382904585
988203585590361740839519681254968145162747098072652
141858607502723046239367209776569259776678871640355
476703121623912558549584789123982974129958278704985
390744483577662104085231708340232525122368990013542
7999613293720681684955293128811292981 / 22458231406
216094878202976126790054324698816432478447511802089
665363641250066501433769538474807742947270581109819
674675916341254734148663444090199254276142009850182
419444726060661342077926179045344110704705488623957
680809306210269199637837088757430354530277343135738
809521441456
```

Snap rounding



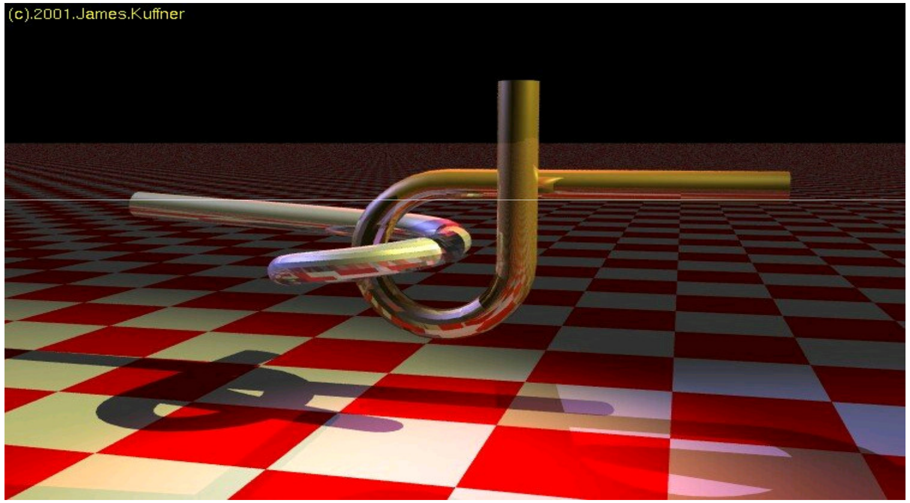
while using limited bit-size coordinates,
snap rounding has nice preservation properties:
geometric and topological

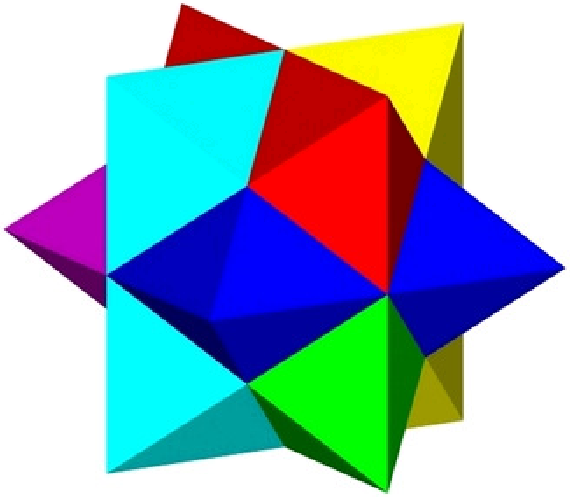
(Snap) Rounding in the course

- snap rounding arrangements of segments: properties and basic algorithm
- improved algorithms
- improved rounding

- rounding in 3D: current state and challenges

Topic IV: Movable separability and assembly planning





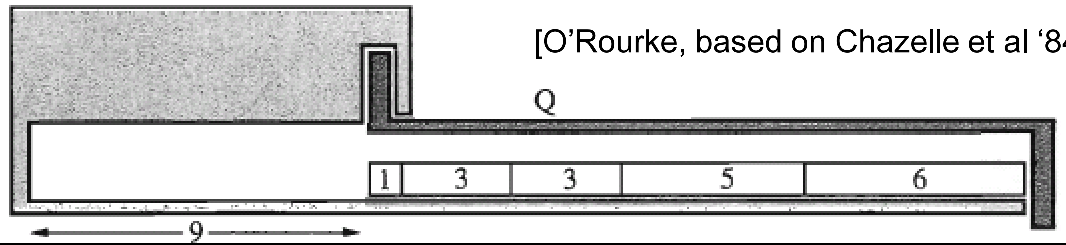
mechanical assembly planning, video

assembly planning guidance

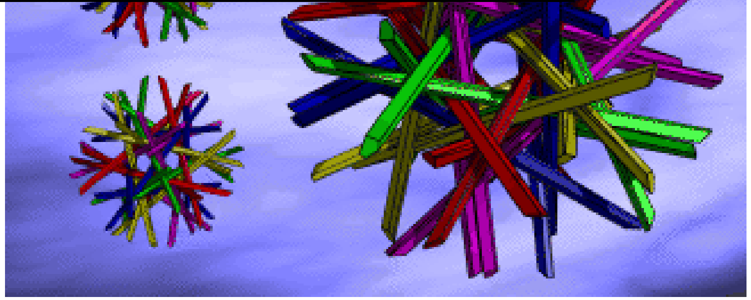
A variety of movable separability problems

- interlocked polygons
- example of a hard separability problem for polygons
- c
- p

video



[O'Rourke, based on Chazelle et al '84]



[Snoeyink-Stolfi 93]

Movable separability in the course

- separation sequences for convex objects in 2D,3D
- 2-handed assembly planning, non-directional blocking graphs, and motion-space; infinite translations in the plane
- improved algorithm: infinitesimal motions
- practice: infinit. motions, infinite translations in 3D

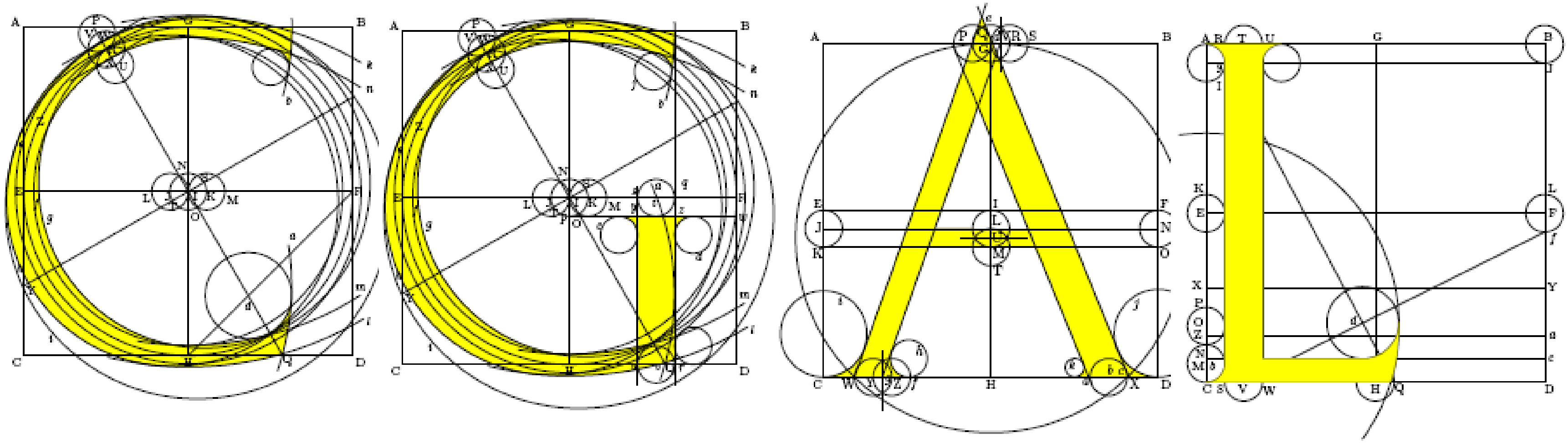
- tolerancing, sensitivity analysis
- assembly planning with more complex motions
- optimization

- Background
- The main topics
- Course mechanics
- Additional topics

Administrivia

- grade:
  20% multiple-choice exam
  80% assignments (one large-scale)
- helpdesk: Monday 1500 – 1600, ACG lab:
  Efi Fogel (~efif) and Eric Berberich (~ericb)
- office hours: Monday 1900 – 2000,
  Schreiber 219

- Background
- The main topics
- Course mechanics
- Additional topics



Computational Geometry Algorithms Library

project goal (1996): *"make the large body of geometric algorithms developed in the field of Computational Geometry available for use in academia and industry“*

The CGAL project and library

- technical criteria: robustness, efficiency, ease of use, homogeneity

- strong connection to ongoing research:
  engineering geometric algorithms =
  algorithm engineering +
  robust geometric computing

The CGAL project in numbers

500,000 lines of C++ code

10,000 downloads/year (+ Linux distributions)

3,500 manual pages

3,000 subscribers to cgal-announce

1,000 subscribers to cgal-discuss

120 packages

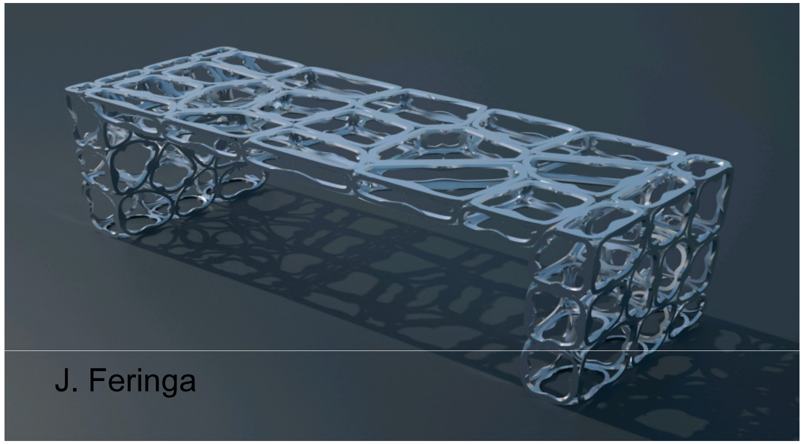
60 commercial users

20 active developers

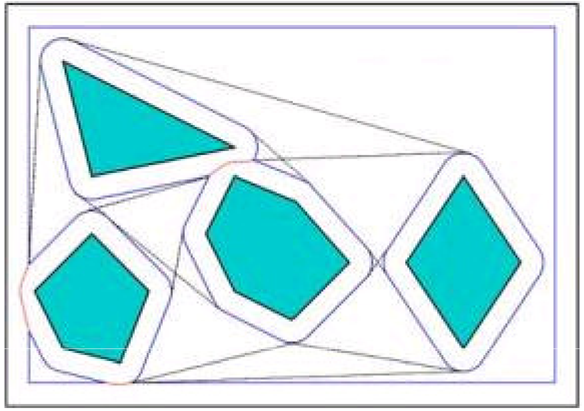
12 months release cycle

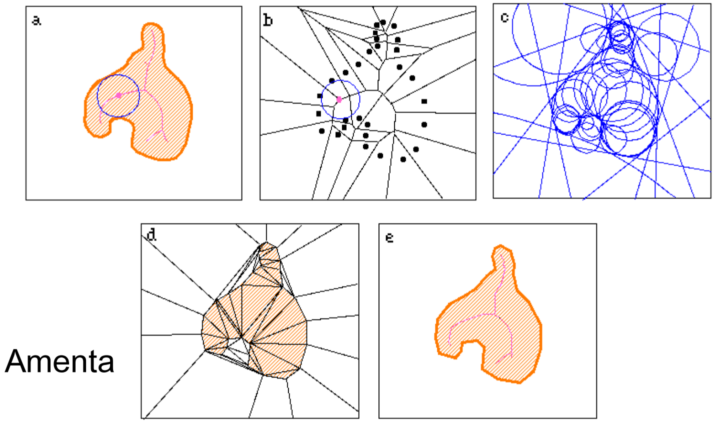
2 licenses: Open Source and commercial

Delaunay triangulations and their relatives as modeling tools

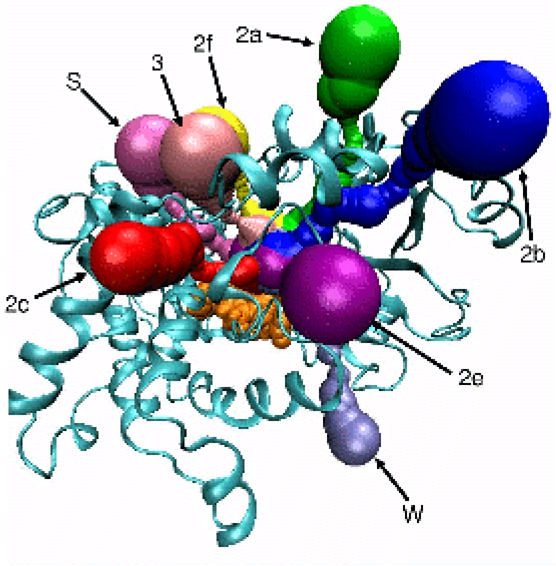


J. Feringa

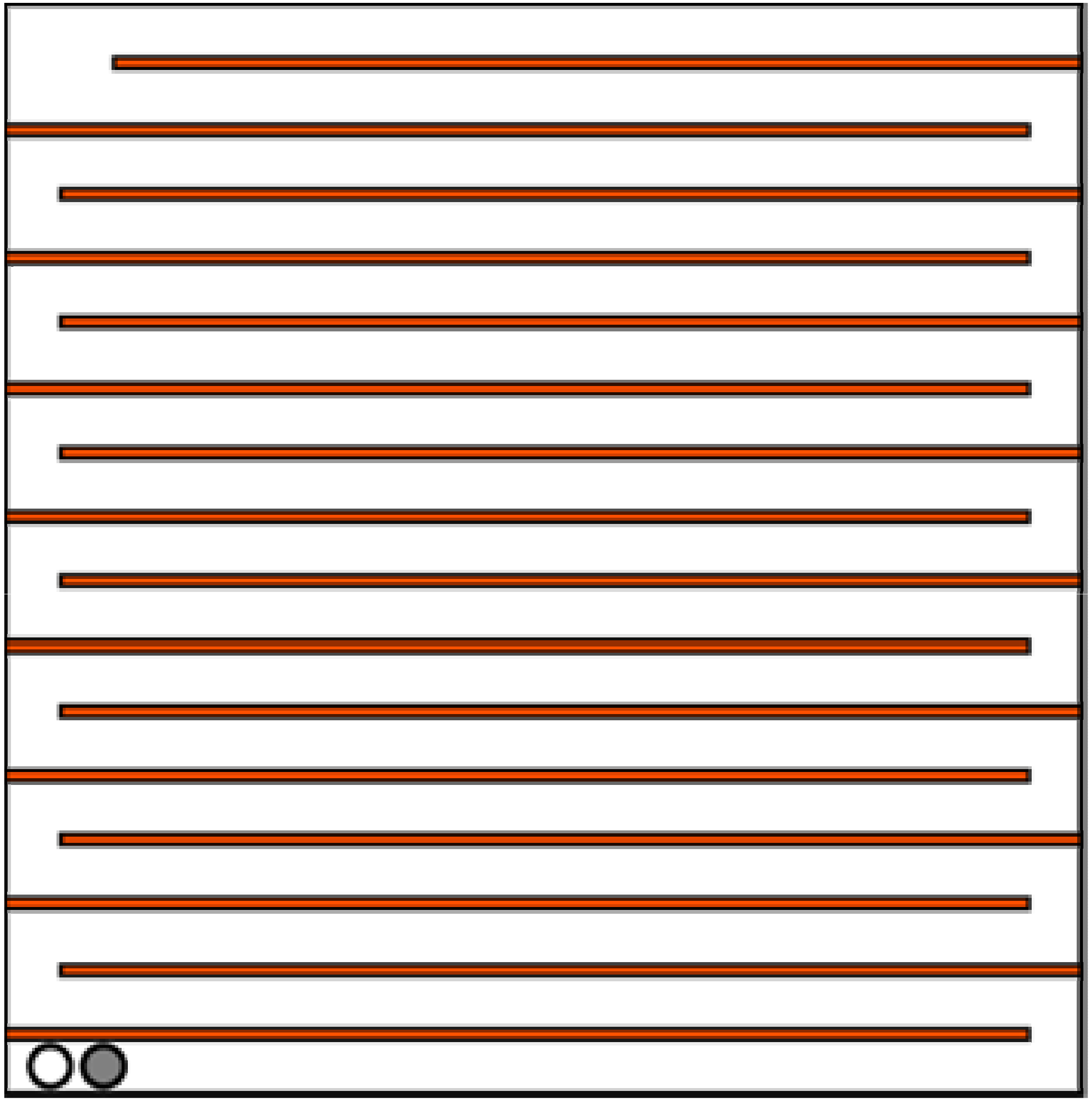




N. Amenta



Algorithmic motion planning



THE END