Robotics and Animatronics in Disney

Lecture 2: Geometric Algorithms for Robotics

Disney Research Pittsburgh

Katsu Yamane
kyamane@disneyresearch.com
Yu Zheng
yu.zheng@disneyresearch.com

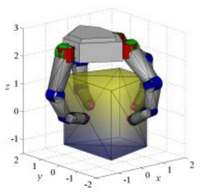
Goals

- Introduce geometry-based algorithms and their application in robotics
- Convince you that they are compelling alternative to numerical algorithms

Disney Research, Pittsburgh

Geometric Problems in Robotics

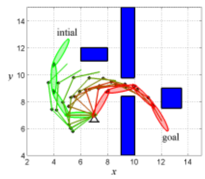
- Grasping / locomotion
  – Feasible total contact wrench is a convex set in 6D space
  – Contact force optimization
  – Grasp quality

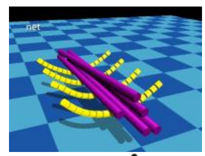


Disney Research, Pittsburgh

Geometric Problems in Robotics

- Collision detection / distance computation
  – Most algorithms deal with polygon models
  – Lose global shape information
  – Large data for accurate representation
  – CAD software work with parametric surfaces (NURBS etc.)





Disney Research, Pittsburgh

Outline

- Contact force optimization in grasping
  – Basic concepts for geometry-based algorithms
  – Introduction to ray-shooting algorithms
- Grasp evaluation
  – Compute the largest inclusion in 6D convex set
- Distance computation
  – Generalized distance
  – Compute the minimum scaling factor for two convex objects to touch

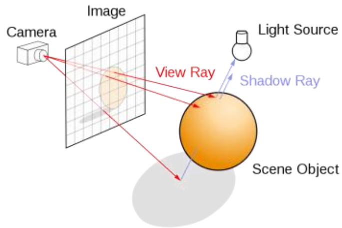
Disney Research, Pittsburgh

Contact Force Optimization with Ray-Shooting

[Zheng and Yamane 2012]

Disney Research Pittsburgh

Ray-Shooting in CG



[Agarwal and Matoušek, SoTC92; Szirmay-Kalos et al., SCCG02]

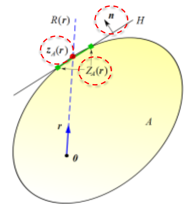
Disney Research, Pittsburgh

General Ray-Shooting Problem

In n-dimensional space
Given: A— a compact convex set
r— an arbitrary nonzero vector
Ray: $R(r) \triangleq \{\lambda r \in \mathbb{R}^n | \lambda \geq 0\}$

Compute:

- $z_A(r)$— farthest intersection point of A with $R(r)$
- $Z_A(r)$— subset of A whose convex hull contains $z_A(r)$
- n— normal of the hyperplane that passes through $z_A(r)$ and supports A



Disney Research, Pittsburgh

Support Mapping and Function

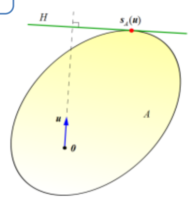
Given: A— a compact convex set
u— an arbitrary nonzero vector

Support mapping— a point in A

$$s_A(u) \triangleq \arg\max_{a \in A} u^T a$$

Support function— a scalar

$$h_A(u) \triangleq \max_{a \in A} u^T a$$



Supporting hyperplane— hyperplane H that passes through $s_A(u)$ with normal u

Disney Research, Pittsburgh

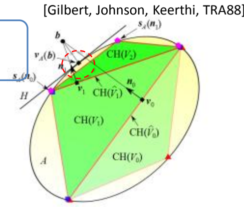
GJK Distance Algorithm

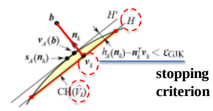
[Gilbert, Johnson, Keerthi, TRA88]

Given: A— a compact convex set
b— a point outside A

Compute:

- $d_A(b)$— minimum distance between b and A
- $v_A(b)$— closest point in A to b
- $V_A(b)$— subset of A whose convex hull contains $v_A(b)$
- H— supporting hyperplane of A at $v_A(b)$

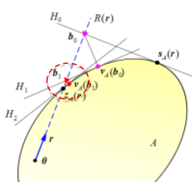


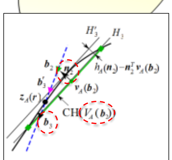


Disney Research, Pittsburgh

GJK-Based Ray-Shooting Algorithm

- $z_A(r)$— farthest intersection point of A with $R(r)$
- $Z_A(r)$— subset of A whose convex hull contains $z_A(r)$
- n— normal of the hyperplane that passes through $z_A(r)$ and supports A





Disney Research, Pittsburgh

ZC Distance Algorithm

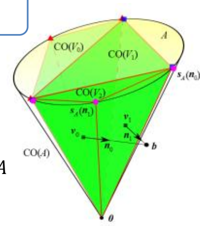
[Zheng and Chew, TRO09]

Given: A— a compact convex set
b— a point

$CO(A)$— convex cone of A

Compute:

- $d_{CO(A)}(b)$— minimum distance between b and $CO(A)$
- $v_{CO(A)}(b)$— closest point in $CO(A$ to b
- $V_{CO(A)}(b)$— subset of A whose convex cone contains $v_{CO(A)}(b)$



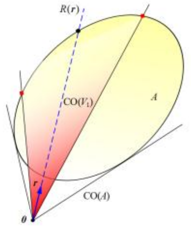
Disney Research, Pittsburgh

ZC Distance Algorithm

$b \in CO(A)$: $d_{CO(A)}(b) = 0, v_{CO(A)}(b) = b, b \in CO\left(V_{CO(A)}(b)\right)$

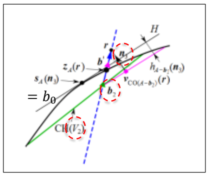
Run ZC Algorithm with $b = r$

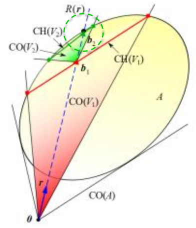
$\rightarrow r \in CO\left(V_{CO(A)}(r)\right)$



ZC-Based Ray-Shooting Algorithm

- $z_A(r)$ ← farthest intersection point of A with $R(r)$
- $Z_A(r)$ ← subset of A whose convex hull contains $z_A(r)$
- n ← normal of the hyperplane that passes through $z_A(r)$ and supports A

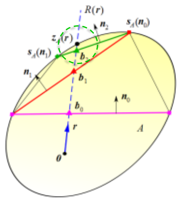


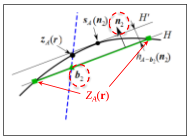


Internal Expanding (IE)

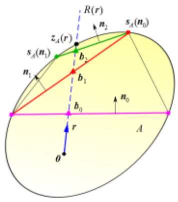
[Zheng, Lin, and Manocha, ICRA10]

- $z_A(r)$ ← farthest intersection point of A with $R(r)$
- $Z_A(r)$ ← subset of A whose convex hull contains $z_A(r)$
- n ← normal of the hyperplane that passes through $z_A(r)$ and supports A

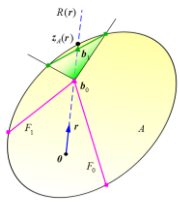




Hybrid Use: ZC-IE

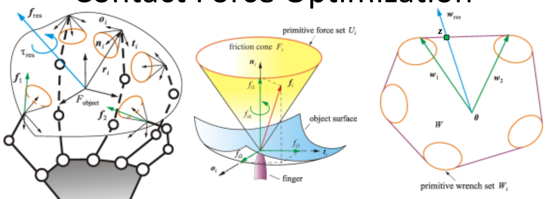


IE iteration



ZC iteration

Contact Force Optimization



$$W_i = \mathbf{G}_i(U_i)$$

3D object space → 6D wrench space

Compute $\mathbf{f}_i \in F_i,\ i = 1, 2, \ldots, m$ such that $\mathbf{w}_{\text{res}} = \sum_{i=1}^{m} \mathbf{G}_i \mathbf{f}_i = -\mathbf{w}_{\text{ext}}$

minimize $\sigma_{L_1} \triangleq \sum_{i=1}^{m} f_{i1}$ $\quad W_{L_1} \triangleq \mathrm{CH}\left(\bigcup_{i=1}^{m} W_i\right)$ $\quad \sigma_{L_1} = \|\mathbf{w}_{\text{res}}\| / \|\mathbf{z}_{W_{L_1}}(\mathbf{w}_{\text{res}})\|$

or $\sigma_{L_\infty} \triangleq \max_{i=1,2,\ldots,m} f_{i1}$ $\quad W_{L_\infty} \triangleq \mathrm{CH}\left(\bigoplus_{i=1}^{m} W_i\right)$ $\quad \sigma_{L_\infty} = \|\mathbf{w}_{\text{res}}\| / \|\mathbf{z}_{W_{L_\infty}}(\mathbf{w}_{\text{res}})\|$

Contact Force Optimization: test 1

TEST 1: Accuracy

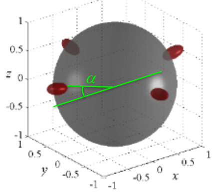
- grasp a sphere against the gravity G
- four contacts have the same elevation angle α on the sphere

Ground Truth

1. No feasible contact forces if $\alpha \geq \tan^{-1}\mu$
2. The minimum values are

$$\sigma_{L_1} = \frac{G}{\mu \cos\alpha - \sin\alpha}$$

$$\sigma_{L\_\infty} = \frac{G}{4(\mu \cos\alpha - \sin\alpha)}$$



Gradually increase α to the limit by setting $\alpha = (1-\lambda)\tan^{-1}\mu$ with $\lambda = 10^{-1}, 10^{-2}, \ldots, 10^{-6}$

Contact Force Optimization: test 1

Relative error of each algorithm to minimize σ_{L_1}

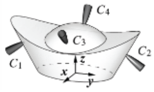
λ	GJK-based	IE	ZC-based	Bi-GJK	ZC-IE	Active-Set
10^{-1}	6.86×10^{-16}	4.93×10^{-6}	0	0	0	3.63×10^{-9}
10^{-2}	1.35×10^{-14}	1.98×10^{-6}	0	1.83×10^{-16}	0	1.65×10^{-6}
10^{-3}	1.38×10^{-13}	5.38×10^{-6}	6.88×10^{-14}	6.91×10^{-14}	6.88×10^{-14}	2.88×10^{-8}
10^{-4}	0	2.84×10^{-6}	6.89×10^{-13}	6.89×10^{-13}	6.89×10^{-13}	3.92×10^{-3}
10^{-5}	6.89×10^{-12}	infeasible	infeasible	infeasible	infeasible	1.09×10^{-1}
10^{-6}	infeasible	infeasible	infeasible	infeasible	infeasible	infeasible

Relative error of each algorithm to minimize σ_{L_∞}

λ	GJK-based	IE	ZC-based	Bi-GJK	ZC-IE	Active-Set
10^{-1}	0	6.53×10^{-6}	0	0	0	7.50×10^{-9}
10^{-2}	0	5.49×10^{-6}	0	6.96×10^{-15}	0	1.18×10^{-8}
10^{-3}	1.38×10^{-13}	5.19×10^{-6}	1.38×10^{-13}	1.38×10^{-13}	1.38×10^{-13}	5.63×10^{-6}
10^{-4}	0	5.39×10^{-6}	0	0	0	9.65×10^{-5}
10^{-5}	1.38×10^{-11}	infeasible	infeasible	infeasible	infeasible	3.00×10^{-4}
10^{-6}	1.50×10^{-16}	infeasible	infeasible	infeasible	infeasible	1.70×10^{-3}

Contact Force Optimization: test 2

TEST 2: Efficiency

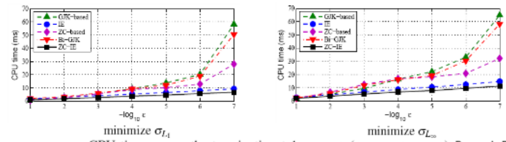


Compute the minimum contact forces for 1000 random w_{res}

Average CPU running time (in milliseconds) and number of iterations (rounded up to the next highest integer in parentheses) of each algorithm for contact force optimization

	GJK-based	IE	ZC-based	Bi-GJK	ZC-IE	Active-Set
minimize σ_{L_1}	13.80 (4)	6.50 (33)	10.22 (26)	12.11 (8)	4.57 (30)	38.94 (24)
minimize σ_{L_∞}	22.23 (4)	10.55 (54)	18.27 (47)	19.25 (10)	7.93 (53)	47.36 (31)

$\varepsilon_{GJK} = \varepsilon_{ZC} = \varepsilon = 10^{-5}$



CPU time versus the termination tolerance ε ($\varepsilon_{GJK} = \varepsilon_{ZC} = \varepsilon$)

Summary

- Four geometry-based algorithms for solving the ray-shooting problem
  - GJK-based
  - ZC-based
  - Internal Expanding (IE)
  - ZC-IE hybrid
- Efficient and accurate contact force optimization by ray-shooting and geometry-based algorithms

Grasp Evaluation

[Zheng and Yamane ICRA 2013]

Disney Research Pittsburgh

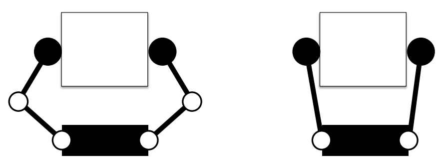
Grasp Quality Measures

- Singular values of the grasp matrix [Li and Sastry, TRA88]
  does not consider friction constraint and force closure
- Geometry of contact polygons [Chinellato et al.; Ponce et al.]
  lacks clear physical meanings
- Size of force-closure contact regions [Nguyen, IJRR88]
  evaluates the safety margin of a force closure
- Grasp force efficiency (GFE): worst-case maximum wrench generated by unit contact forces [Kirkpatrick et al., DCG92; Ferrari and Canny, ICRA92]
  quantitative evaluation of a grasp beyond force closure

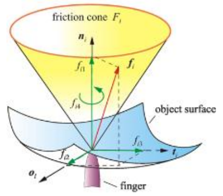
Based solely on contact locations

Fair Evaluation of a Grasp?

Ability to exert contact forces also depends on
- Hand configuration
- Passive joints and joint coupling

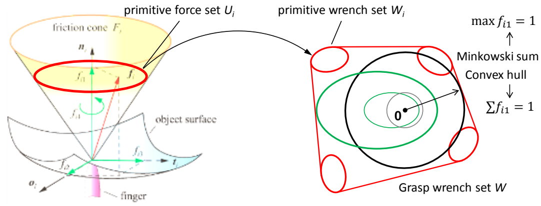


Grasp Force Efficiency (GFE)



3-D object space

Grasp Force Efficiency (GFE)



3-D object space

6-D wrench space

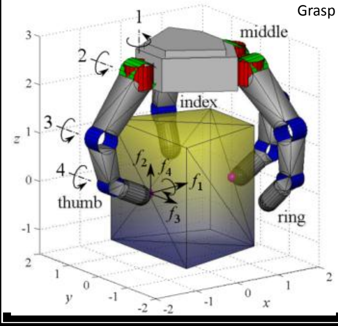
Primitive contact force:
- unit normal component
- on the boundary of friction cone

Grasp wrench set (GWS): consists of all resultant wrenches generated by unit contact forces

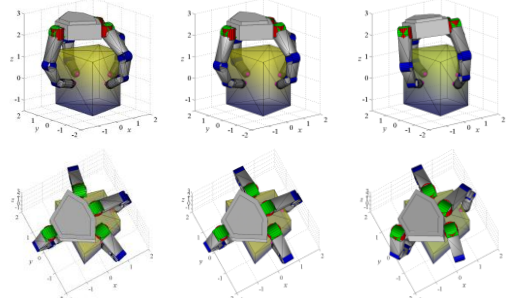
Grasp Force Efficiency (GFE): largest scaling factor of the a convex set to be inscribed in GWS

Example

Soft-finger contacts
Joints 3 and 4 (blue) coupled
Grasp wrench set from Minkowski sum



GFE with Same Contact Locations

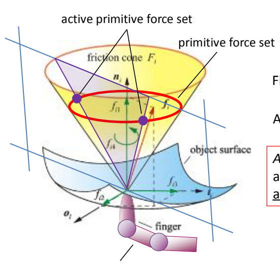


Case 1 $r = 0.5624$

Case 2 $r = 0.5624$

Case 3 $r = 0.5624$

Active Grasp Force Efficiency (A-GFE)



finger mechanism & configuration

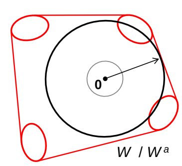
Finger Jacobian: $\tau_i = \mathbf{J}_i^T \mathbf{f}_i$

Active contact force: $\mathbf{f}_i = \mathbf{J}_i^{+T} \tau_i$

Active grasp wrench set W^a: consists of all resultant wrenches generated by unit active contact forces

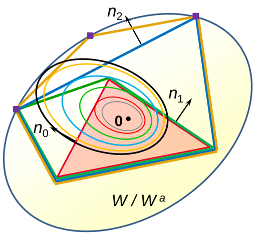
How to Compute (A-)GFE?

- Problem: local minima
- Existing algorithms for GFE: approximate the friction cones by polygonal pyramids

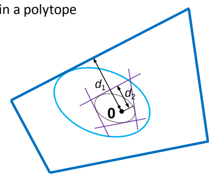


- New algorithm
  - Finds the generalized penetration distance
  - Global optimum without friction cone approximation
  - Applicable to (A-)GFE with convex hull and Minkowski sum
  - Requires only linear algebra calculations

Algorithm



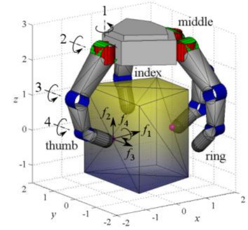
Calculate the largest inscribed set in a polytope



Calculate the minimum of d_1/d_2 over all facets

Disney Research, Pittsburgh

Numerical Results

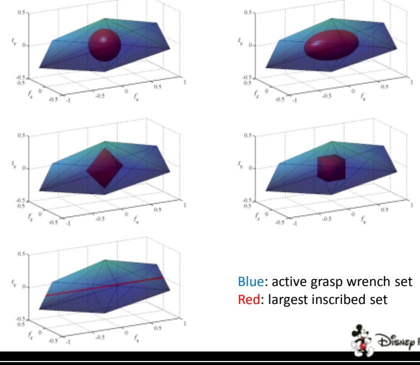


Test 1:
- two fingers
- joint 1 (red) is locked

Result:
- W is 6-D, but W^a is 3-D

Disney Research, Pittsburgh

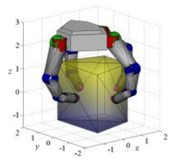
Different Inscribed Sets



Blue: active grasp wrench set
Red: largest inscribed set

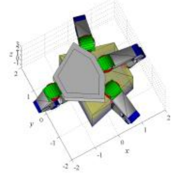
Disney Research, Pittsburgh

Numerical Results



Test 2:
- four fingers
- Joint 1 (red) is locked

Result:
- W is 6-D, but W^a is 5-D



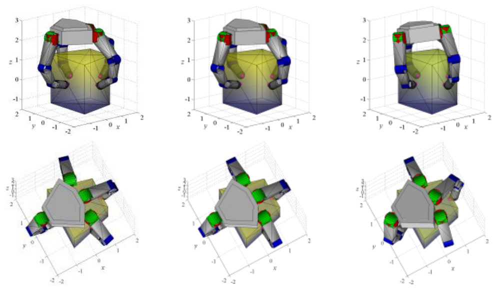
Test 3:
- no joint is locked

Result:
- both W and W^a are 6-D
- r = 0.5624, but r^a = 0.3693

Disney Research, Pittsburgh

Numerical Results

Test 4: same contact locations, different hand configurations

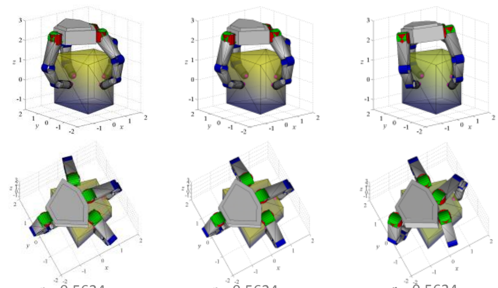


r = 0.5624 | r = 0.5624 | r = 0.5624

Disney Research, Pittsburgh

Numerical Results

Test 4: same contact locations, different hand configurations



r = 0.5624, r^a = 0.3693 | r = 0.5624, r^a = 0.0395 | r = 0.5624, r^a = 0.1536

Disney Research, Pittsburgh

Summary

- Proposed the Active Grasp Force Efficiency (A-GFE) measure that considers hand mechanism/configuration
- Developed an accurate and efficient algorithm for computing the generalized penetration distance, including the (A-)GFE measure

Disney Research, Pittsburgh

Generalized Distance

[Zheng and Yamane ICRA 2013]

Disney Research Pittsburgh

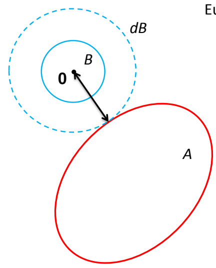
Distance Functions

- Euclidean distance
  - Separation case [Gilbert et al., TRA88,90; Lin and Canny, ICRA91]
  - Penetration case [Zheng and Yamane, ICRA13]
- Generalized distance [Zhu et al., TRA03,04]
  - Separation [this paper]
  - Penetration [Zheng and Yamane, ICRA13]
- Growth distance [Ong and Gilbert, TRA96; Zheng and Yamane, WAFR12]
- Distance in configuration space [LaValle 2006, Zhang et al., SPM06]

Disney Research, Pittsburgh

Definitions of Distance

Given: *A* — a compact convex set not containing the origin **0**



Euclidean distance between **0** and *A*

B: unit ball

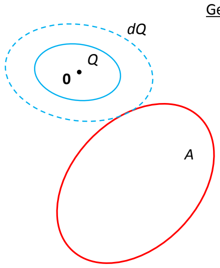
d: minimum scaling factor such that *dB* and *A* are not disjoint

Separation [Gilbert et al., TRA88,90; Lin and Canny, ICRA91]
Penetration [Zheng and Yamane, ICRA13]

Disney Research, Pittsburgh

Definitions of Distance

Given: *A* — a compact convex set not containing the origin **0**



Generalized distance between **0** and *A*

Q: a “guage set”

d: minimum scaling factor such that *dQ* and *A* are not disjoint

↓

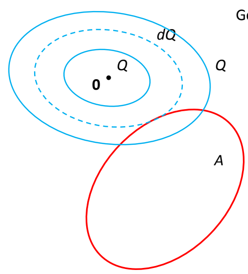
Generalized distance [Zhu et al., TRA03,04]

Separation [this paper]
Penetration [Zheng and Yamane, ICRA13]

Disney Research, Pittsburgh

Definitions of Distance

Given: *A* — a compact convex set not containing the origin **0**



Generalized distance between convex objects

Q: another convex object

d: minimum scaling factor such that *dQ* and *A* are not disjoint

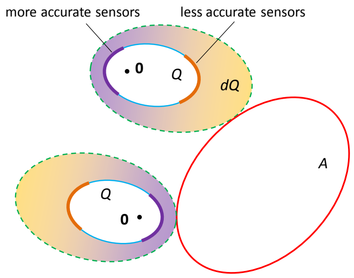
↓

d>1: separation

d<1: penetration

Disney Research, Pittsburgh

Application in Motion Planning

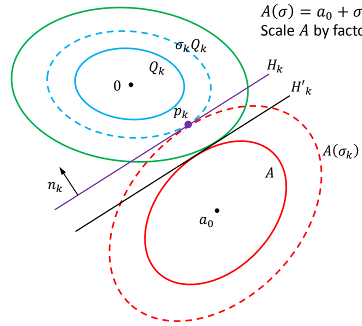


A — an obstacle Q — a robot

Algorithm Overview

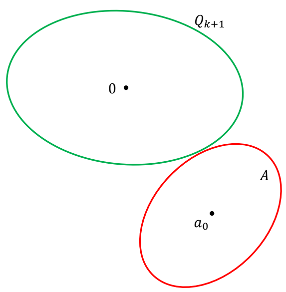
$A(\sigma) = a_0 + \sigma(A - a_0)$

Scale A by factor σ around a_0



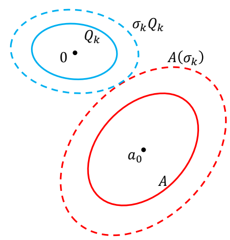
Ray-shooting [Zheng and Yamane, WAFR2012]

Algorithm Overview



Stopping criterion: $\sigma_k - 1 < \varepsilon$

Computing σ_k and n_k



$A(\sigma) = a_0 + \sigma(A - a_0)$

$$\sigma_k = \min_{\boxed{A(\sigma)\cap\sigma Q_k \neq \emptyset}, \sigma \geq 0} \sigma$$

$\Updownarrow$

$$0 \in A(\sigma) - \sigma Q_k$$

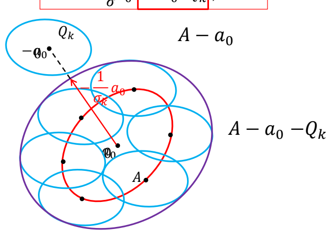
$\Updownarrow$

$$-\frac{1}{\sigma} a_0 \in A - a_0 - Q_k$$

$$\boxed{\sigma_k = \min_{-\frac{1}{\sigma}a_0 \in A - a_0 - Q_k, \sigma \geq 0} \sigma}$$

Computing σ_k and n_k

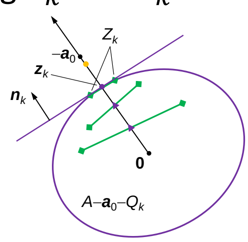
$$\boxed{\sigma_k = \min_{-\frac{1}{\sigma}a_0 \in \boxed{A - a_0 - Q_k}, \sigma \geq 0} \sigma}$$



Finding $-\frac{1}{\sigma_k}$: ray-shooting [Zheng and Yamane, WAFR2012]

Computing σ_k and n_k

1) $z_k \rightarrow \sigma_k = \frac{\|a_0\|}{\|z_k\|}$



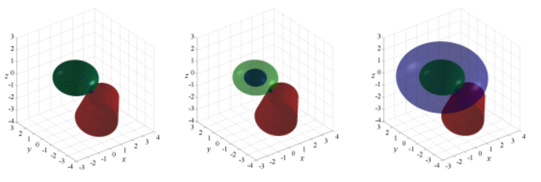
[Zheng and Yamane, WAFR2012]

2) Z_k: set of points that represent z_k as their convex combination

$\rightarrow n_k$: normal of the supporting hyperplane

Accuracy and Speed

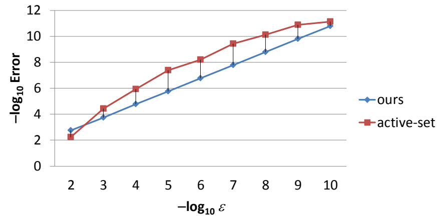
Red — *A*　　Blue — *Q*　　Green — *dQ*



contact $d = 1$　　separate $d = 2$　　penetrate $d = 0.5$

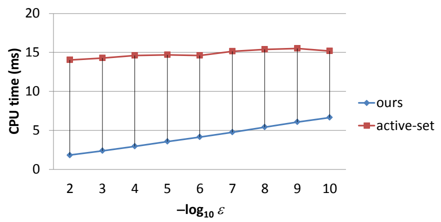
3000 random pairs for each case

Error



The error of our algorithm is linear with the termination tolerance.
Our algorithm is as accurate as the active-set algorithm by MATLAB.

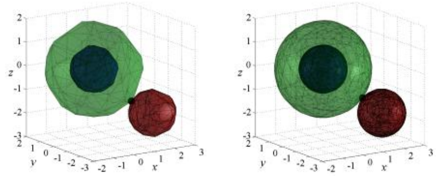
CPU Running Time



Our algorithm is faster than the active-set algorithm by MATLAB.

n-Dimensional Polytopes

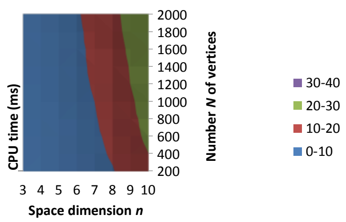
A and *Q* are *n-dimensional* polytopes with *N* vertices each on a unit ball



200 vertices for each　　1000 vertices for each

3-D illustration

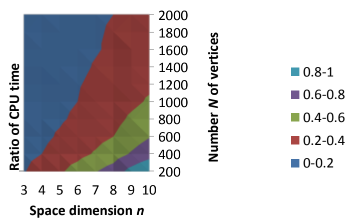
CPU Running Time



The CPU running time increases slowly along with the space dimension and the number of vertices.

Comparison with the Simplex Method

$$\frac{\text{CPU time of our algorithm}}{\text{CPU time of the Simplex method}}$$



Our algorithm is much faster than the Simplex method by MATLAB especially in low-dimensional space with a large number of points.

Summary

- Generalized distance: minimum scaling factor for two convex objects to touch
- A geometry-based algorithm for efficient computation of the generalized distance

Discussion

- Advantages of geometry-based algorithms
  - Global optimum
  - Fast
  - Consists of a few basic geometric computation
  - Accuracy comparable to standard numerical optimization
- Applications
  - Planning
  - Contact simulation
  - Grasp analysis