Shortest Paths in Graphs of Convex Sets

and their applications

Pablo A. Parrilo

Laboratory for Information and Decision Systems (LIDS)
Massachusetts Institute of Technology
parrilo@mit.edu

ICCOPT/MOPTA
Lehigh University
July 27, 2022

Wonderful coauthors

Tobia Marcucci

Jack Umenberger

Russ Tedrake

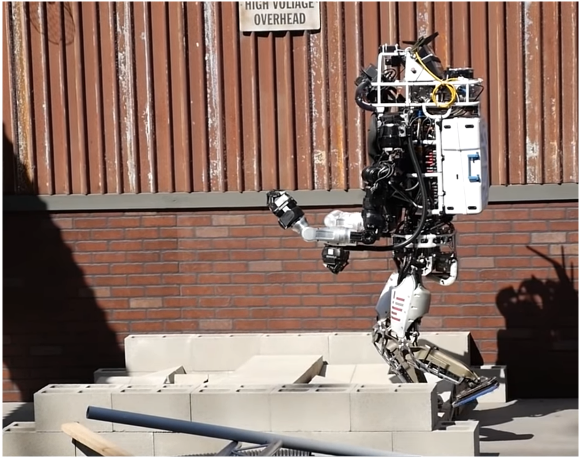
References

- Marcucci, Umenberger, Parrilo, Tedrake "Shortest paths in graphs of convex sets," arXiv:2101.11565

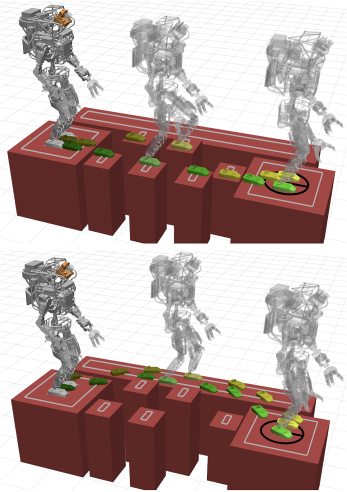
Outline

- Motivation (for us): robotics and planning
- Graphs of Convex Sets framework
- Shortest paths on GCS
- Convex relaxation and mixed-integer formulation
- Numerical results and implementation
- Conclusions and future directions

Footstep planning for legged robots

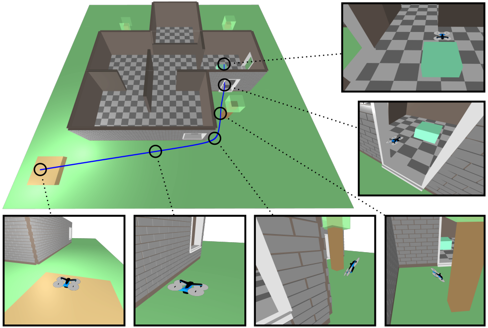


[MIT DRC (DARPA Robotics Challenge) Team]



[Deits and Tedrake '14]

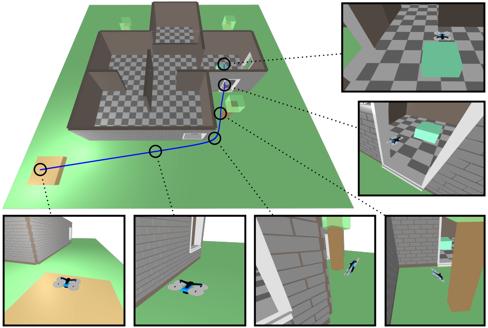
Motion planning



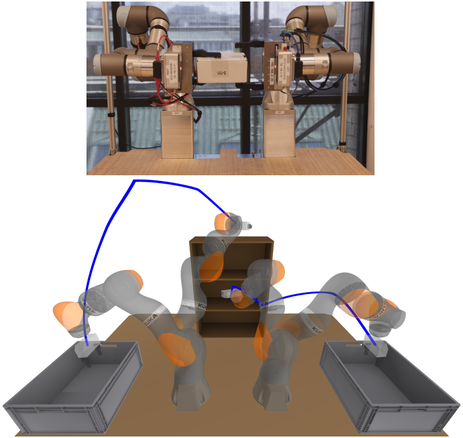
Drone navigation in 3D environments

[1] Marcucci, Petersen, von Wrangel, Tedrake "Motion planning around obstacles with convex optimization" (2022)

Motion planning



Drone navigation in 3D environments



Manipulation with multiple robots/tasks

[1] Marcucci, Petersen, von Wrangel, Tedrake "Motion planning around obstacles with convex optimization" (2022)

State of the art in robot motion planning

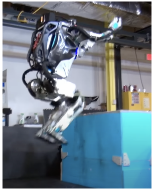
Atlas motion planning [1,2]

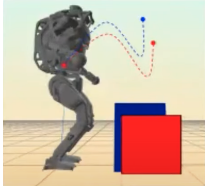
Offline:

- Library of template behaviors

Online:

- Continuous blending of behaviors
- Parametric convex optimization





[1] Kuindersma "Recent progress on Atlas, the world's most dynamic humanoid robot" (MIT Robotics Today 2020)

[2] Deits "Making Atlas Dance, Run, and Jump" (6th Workshop on Legged Robots, ICRA 2022)

State of the art in robot motion planning

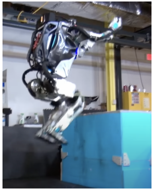
Atlas motion planning [1,2]

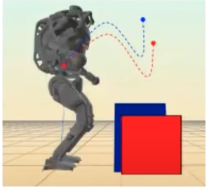
Offline:

- Library of template behaviors

Online:

- Continuous blending of behaviors
- Parametric convex optimization





- The discrete sequence of behaviors is hand-designed
- The robot can't react to structural changes in the environment

[1] Kuindersma "Recent progress on Atlas, the world's most dynamic humanoid robot" (MIT Robotics Today 2020)

[2] Deits "Making Atlas Dance, Run, and Jump" (6th Workshop on Legged Robots, ICRA 2022)

Motivation

- Interesting, challenging, and practically relevant problems
- Both continuous and combinatorial features
- Moderately high dimension ($10^1 - 10^4$ variables)
- Typically, nontrivial dynamical constraints

Motivation

- Interesting, challenging, and practically relevant problems
- Both continuous and combinatorial features
- Moderately high dimension ($10^1 - 10^4$ variables)
- Typically, nontrivial dynamical constraints

Goal: An optimization-friendly abstraction to best capture their essence

Motivation

- Interesting, challenging, and practically relevant problems
- Both continuous and combinatorial features
- Moderately high dimension ($10^1 - 10^4$ variables)
- Typically, nontrivial dynamical constraints

Goal: An optimization-friendly abstraction to best capture their essence

(**Goal 2:** Significantly improve over the state of the art!)

Motivation

- Interesting, challenging, and practically relevant problems
- Both continuous and combinatorial features
- Moderately high dimension ($10^1 - 10^4$ variables)
- Typically, nontrivial dynamical constraints

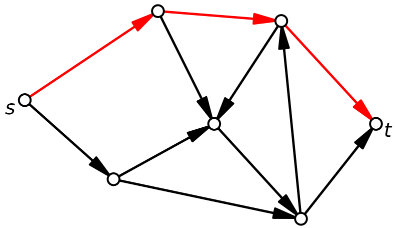
Goal: An optimization-friendly abstraction to best capture their essence

(**Goal 2:** Significantly improve over the state of the art!)

Our proposal: *Shortest paths in Graphs of Convex Sets (GCS)*

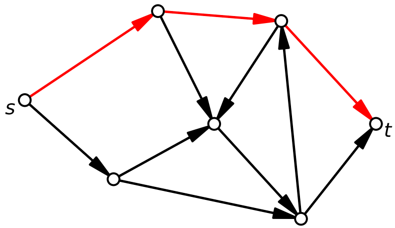
Big picture

Shortest-Path Problem (SPP)

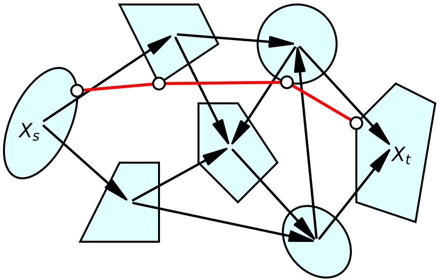


Big picture

Shortest-Path Problem (SPP)

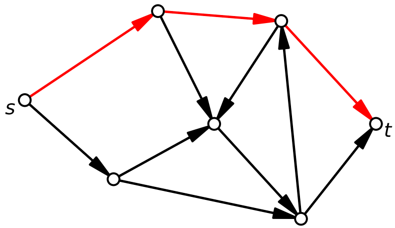


SPP in Graphs of Convex Sets (GCS)

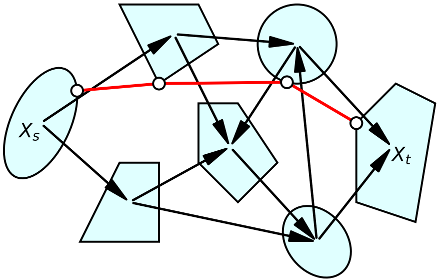


Big picture

Shortest-Path Problem (SPP)



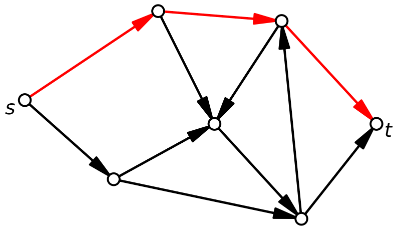
SPP in Graphs of Convex Sets (GCS)



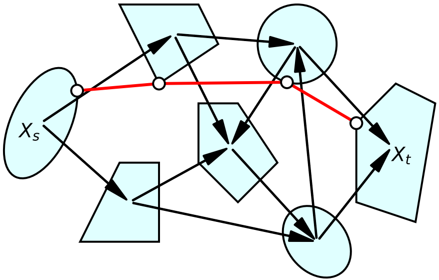
- Very versatile problem formulation

Big picture

Shortest-Path Problem (SPP)



SPP in Graphs of Convex Sets (GCS)



- Very versatile problem formulation
- Efficiently solvable in practice (although NP-hard)
  - Mixed-integer formulation with very tight convex relaxation

GCS gracefully blends discrete and continuous

Why? Highlights interactions and feedback between:

Structure / Whole

- High-level decisions
- Discrete "combinatorial skeleton" of trajectory/motion (e.g. obstacle avoidance)

Details / Parts

- Lower-level optimization considerations (fuel, time, cost, etc)
- Typically continuous and "nice".

GCS gracefully blends discrete and continuous

Why? Highlights interactions and feedback between:

Structure / Whole

- High-level decisions
- Discrete "combinatorial skeleton" of trajectory/motion (e.g. obstacle avoidance)

Details / Parts

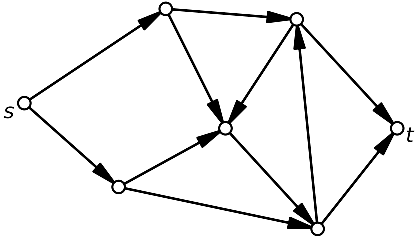
- Lower-level optimization considerations (fuel, time, cost, etc)
- Typically continuous and "nice".

Useful guiding principle

"Easy" problems should remain easy.

SPP in a graph of convex sets

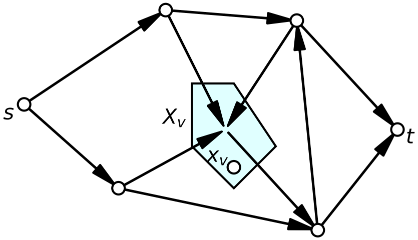
- Directed Graph $G = (V, E)$



Joint optimization over paths AND vertex locations!

SPP in a graph of convex sets

- Directed Graph $G = (V, E)$
- For each vertex $v \in V$ we have
  - Convex set X_v
  - Point $x_v \in X_v$



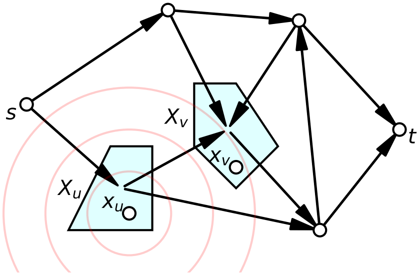
Joint optimization over paths AND vertex locations!

SPP in a graph of convex sets

- Directed Graph $G = (V, E)$
- For each vertex $v \in V$ we have
  - Convex set X_v
  - Point $x_v \in X_v$
- Edge $e = (u, v) \in E$ has convex "length"

$$\ell_e : X_u \times X_v \to \mathbb{R}_{\geq 0} \cup \{\infty\}$$

  - Can enforce convex constraints $(x_u, x_v) \in X_e$



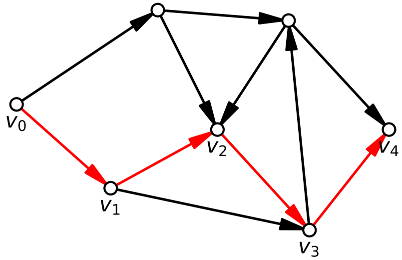
Joint optimization over paths AND vertex locations!

SPP in a graph of convex sets

- Directed Graph $G = (V, E)$
- For each vertex $v \in V$ we have
  - Convex set X_v
  - Point $x_v \in X_v$
- Edge $e = (u, v) \in E$ has convex "length"

$$\ell_e : X_u \times X_v \to \mathbb{R}_{\geq 0} \cup \{\infty\}$$

  - Can enforce convex constraints $(x_u, x_v) \in X_e$
- A path π is a
  - Sequence of distinct vertices $(v_k)_{k=0}^{K}$
  - $v_0 = s$ and $v_K = t$
  - $(v_k, v_{k+1}) \in E$



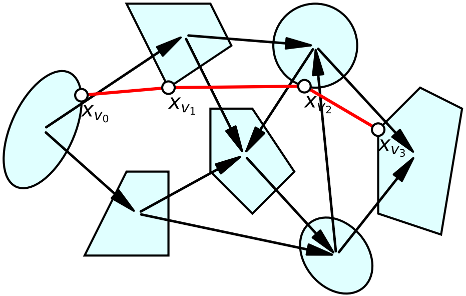
Joint optimization over paths AND vertex locations!

SPP in a graph of convex sets

- Directed Graph $G = (V, E)$
- For each vertex $v \in V$ we have
  - Convex set X_v
  - Point $x_v \in X_v$
- Edge $e = (u, v) \in E$ has convex "length"

$$\ell_e : X_u \times X_v \to \mathbb{R}_{\geq 0} \cup \{\infty\}$$

  - Can enforce convex constraints $(x_u, x_v) \in X_e$
- A path π is a
  - Sequence of distinct vertices $(v_k)_{k=0}^K$
  - $v_0 = s$ and $v_K = t$
  - $(v_k, v_{k+1}) \in E$



Shortest-path problem (GCS)

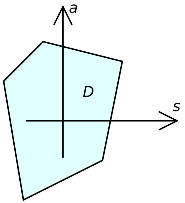
$$\min_{\pi \in \Pi} \min_{x \in X} \sum_{e \in E_\pi} \ell_e(x_u, x_v)$$

Joint optimization over paths AND vertex locations!

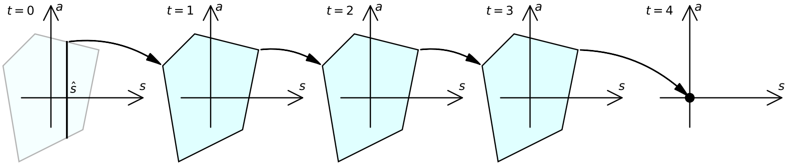
Optimal control as an SPP

Constrained linear regulation

$$\begin{array}{lll} \text{minimize} & \sum_{t=0}^{T-1} c(s_t, a_t) & \\ \text{subject to} & s_{t+1} = As_t + Ba_t, & \forall t \\ & (s_t, a_t) \in D, & \forall t \\ & s_0 = \hat{s}, \ s_T = 0 & \end{array}$$



Shortest-path problem



- Edge lengths

  $$c(s_t, a_t)$$

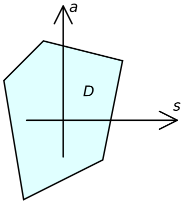
- Edge constraints

  $$s_{t+1} = As_t + Ba_t$$

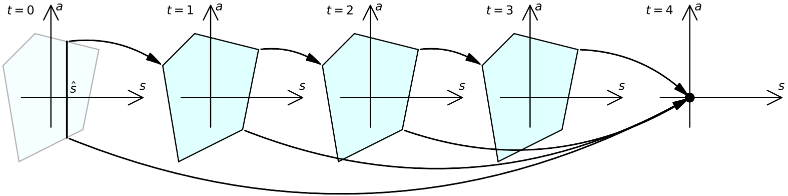
Optimal control as an SPP

Minimum time

$$\begin{aligned} \text{minimize} \quad & T \\ \text{subject to} \quad & s_{t+1} = As_t + Ba_t, \qquad \forall t \\ & (s_t, a_t) \in D, \qquad \forall t \\ & s_0 = \hat{s}, \;\; s_T = 0 \end{aligned}$$



Shortest-path problem



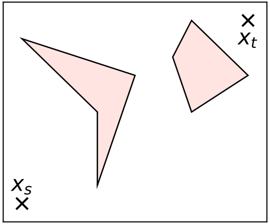
- Edge lengths

  $$1$$

- Edge constraints

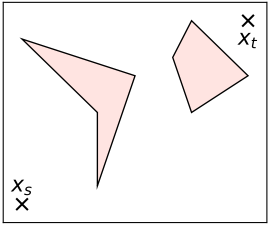
  $$s_{t+1} = As_t + Ba_t$$

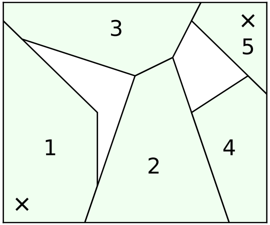
Collision-free motion planning as an SPP



Connect x_s to x_t via a collision-free polygonal line

Collision-free motion planning as an SPP

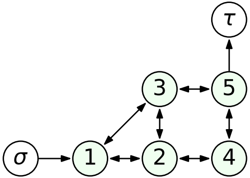


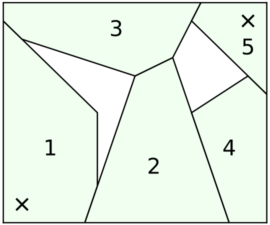


Connect x_s to x_t via a collision-free polygonal line

- Decompose free space in safe convex regions

Collision-free motion planning as an SPP

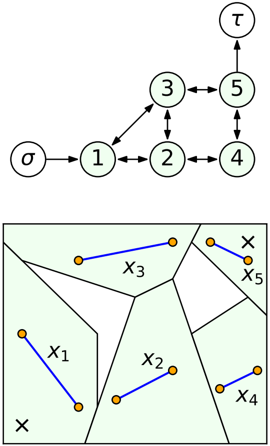




Connect x_s to x_t via a collision-free polygonal line

- Decompose free space in safe convex regions
- Construct adjacency graph

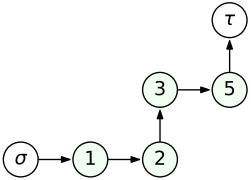
Collision-free motion planning as an SPP

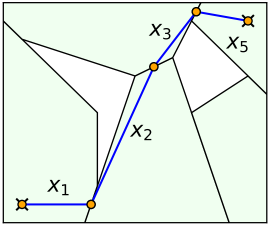


Connect x_s to x_t via a collision-free polygonal line

- Decompose free space in safe convex regions
- Construct adjacency graph
- Assign a line segment to each region
  - Segment $\in$ region $\Leftrightarrow$ start and end point $\in$ region

Collision-free motion planning as an SPP

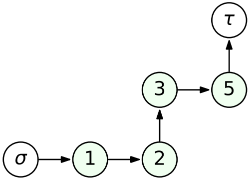


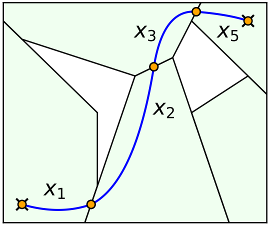


Connect x_s to x_t via a collision-free polygonal line

- Decompose free space in safe convex regions
- Construct adjacency graph
- Assign a line segment to each region
  - Segment $\in$ region $\Leftrightarrow$ start and end point $\in$ region
- Continuity enforced as an edge constraint

Collision-free motion planning as an SPP



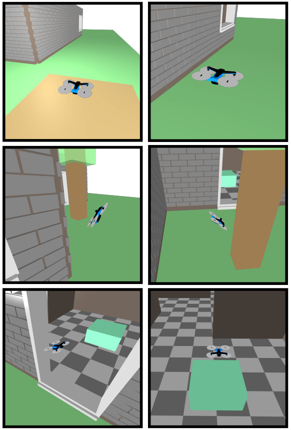


Connect x_s to x_t via a collision-free polygonal line

- Decompose free space in safe convex regions
- Construct adjacency graph
- Assign a line segment to each region
  - Segment $\in$ region $\Leftrightarrow$ start and end point $\in$ region
- Continuity enforced as an edge constraint

- Extends to polynomials using Bézier curves

Collision-free motion planning as an SPP



Connect x_s to x_t via a collision-free polygonal line

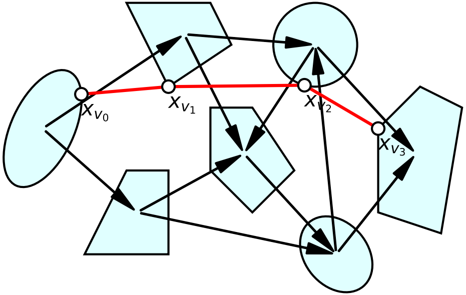
- Decompose free space in safe convex regions
- Construct adjacency graph
- Assign a line segment to each region
  - Segment $\in$ region $\Leftrightarrow$ start and end point $\in$ region
- Continuity enforced as an edge constraint

- Extends to polynomials using Bézier curves
- Takes into account the timing of the trajectory
  - Velocity constraints
  - Continuity of derivatives
  - Trajectory-duration constraints

Complexity?

- Fixing *either* π or x, problem is "easy":
  - Fixed π (sequence): convex optimization problem over x_v
  - Fixed x_v (locations): "standard" shortest path problem
- Unfortunately, NP-hard if we search for both

Bad news, but certainly expected...

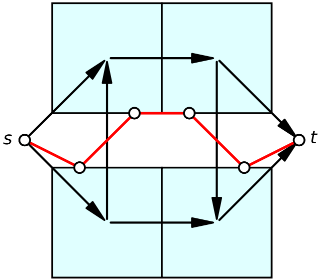


Shortest-path problem (GCS)

$$\min_{\pi \in \Pi} \min_{x \in X} \sum_{e \in E_\pi} \ell_e(x_u, x_v)$$

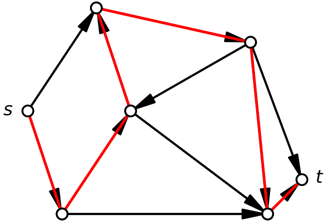
NP-Hardness: reduction from Hamiltonian-path problem

$$\ell_e(x_u, x_v) = \|x_v - x_u\|_2^2$$



"Is there a path that visits each vertex?"

Hamiltonian-path problem



- One of Karp's 21 NP-complete problems
- Reducible to our SPP in polynomial time
  - $\Rightarrow$ SPP/GCS is NP-hard

(Some) Related work

Location/allocation problems (Cooper 1963)

Fermat-Weber, facility location, etc.

Computational geometry

Zookeeper problem, watchman problems, safari problems, art gallery problems, etc.
Typically only 2-3 dimensions, approximation algorithms.

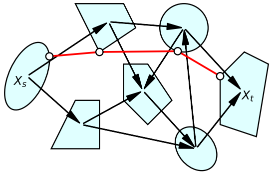
Graph problems with neighborhoods

Variants of the TSP and the MSTP, sometimes with quite strong restrictions on the neighborhoods (e.g., polygonal regions on the 2D plane). Very low dimensional, tackled using mixed-integer nonconvex programming [1,2,3] – not competitive with our approach

---

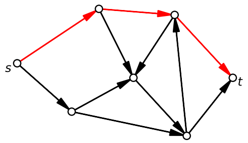
[1] Gentilini et al. "The travelling salesman problem with neighbourhoods: MINLP solution" (Optimization Methods and Software 2013)
[2] Blanco et al. "Minimum spanning trees with neighborhoods: Mathematical programming formulations and solution methods" (Eur. Jour. of OR 2017)
[3] Burdick et al. "From multi-target sensory coverage to complete sensory coverage" (ICRA 2021)

Our solution approach



- A compact mixed-integer convex program
  - With very tight convex relaxation
  - Quite often exact, otherwise do rounding or B&B

Our solution approach

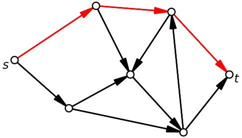


- A compact mixed-integer convex program
  - With very tight convex relaxation
  - Quite often exact, otherwise do rounding or B&B

Construction of the mixed-integer program

1. Integer program of classical SPP
   - Path parameterized using a "flow" variable $y_e \in \{0,1\}$ per edge e
   - Linear constraints enforce conservation of flow

Our solution approach

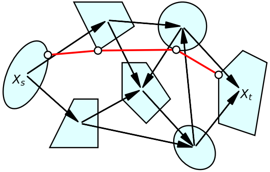


- A compact mixed-integer convex program
  - With very tight convex relaxation
  - Quite often exact, otherwise do rounding or B&B

Construction of the mixed-integer program

1. Linear program of classical SPP
   - Path parameterized using a "flow" variable $y_e \in [0, 1]$ per edge e
   - Linear constraints enforce conservation of flow

Our solution approach

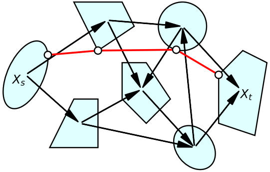


- A compact mixed-integer convex program
  - With very tight convex relaxation
  - Quite often exact, otherwise do rounding or B&B

Construction of the mixed-integer program

1. Linear program of classical SPP
   - Path parameterized using a "flow" variable $y_e \in [0, 1]$ per edge e
   - Linear constraints enforce conservation of flow
2. Natural extension to graph of convex sets
   - Yields bilinear program (products between vertex positions x_v and flows y_e)

Our solution approach



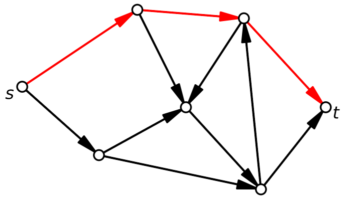
- A compact mixed-integer convex program
  - With very tight convex relaxation
  - Quite often exact, otherwise do rounding or B&B

Construction of the mixed-integer program

1. Linear program of classical SPP
   - Path parameterized using a "flow" variable $y_e \in [0, 1]$ per edge e
   - Linear constraints enforce conservation of flow
2. Natural extension to graph of convex sets
   - Yields bilinear program (products between vertex positions x_v and flows y_e)
3. Set-based perspective convex relaxation of the bilinearities
   - Exact when $y_e \in \{0, 1\}$ for all edges e

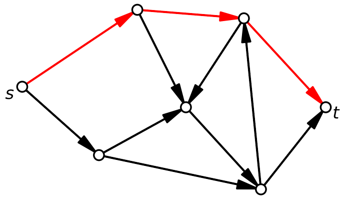
Step 1: Linear program of classical SPP

- Convex hull of paths well understood (flow polytope)
- Edge costs c_e are nonnegative scalars
- Path parameterized by the flows $\varphi_e \in [0, 1]$



Step 1: Linear program of classical SPP

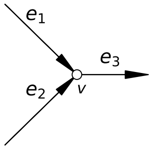
- Convex hull of paths well understood (flow polytope)
- Edge costs c_e are nonnegative scalars
- Path parameterized by the flows $\varphi_e \in [0, 1]$



Min-cost flow LP

$$\begin{array}{ll} \text{minimize} & \sum_{e \in E} c_e \varphi_e \\ \text{subject to} & \varphi_v \in \Phi_v, \qquad \forall v \in V \end{array}$$

- φ_v is vector of flows incident with v
- Φ_v is (local) flow polytope
  - Flows are nonnegative
  - Flow through a vertex is conserved and at most one



$\varphi_v = (\varphi_{e_1}, \varphi_{e_2}, \varphi_{e_3})$

Step 2: Bilinear program

Straightforward extension of the LP

$$\begin{array}{rll} \text{minimize} & \displaystyle\sum_{e=(u,v)\in E} \ell_e(x_u, x_v)\varphi_e & \\ \text{subject to} & \varphi_v \in \Phi_v, \; x_v \in X_v, & \forall v \in V \end{array}$$

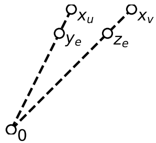
Step 2: Bilinear program

Straightforward extension of the LP

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{e=(u,v)\in E} \ell_e(x_u, x_v)\varphi_e \\ \text{subject to} & \varphi_v \in \Phi_v,\ x_v \in X_v, \qquad \forall v \in V \end{array}$$

Introduce auxiliary variables y_e and z_e

$$\begin{array}{lll} \text{minimize} & \displaystyle\sum_{e\in E} \ell_e\left(\frac{y_e}{\varphi_e}, \frac{z_e}{\varphi_e}\right)\varphi_e & \\ \text{subject to} & \varphi_v \in \Phi_v,\ x_v \in X_v, & \forall v \in V \\ & y_e = \varphi_e x_u,\ z_e = \varphi_e x_v, & \forall e = (u, v) \in E \end{array}$$



Step 2: Bilinear program

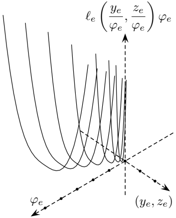
Straightforward extension of the LP

$$\begin{aligned} \text{minimize} \quad & \sum_{e=(u,v)\in E} \ell_e(x_u, x_v)\varphi_e \\ \text{subject to} \quad & \varphi_v \in \Phi_v,\ x_v \in X_v, \qquad \forall v \in V \end{aligned}$$

Introduce auxiliary variables y_e and z_e

$$\begin{aligned} \text{minimize} \quad & \sum_{e\in E} \ell_e\left(\frac{y_e}{\varphi_e}, \frac{z_e}{\varphi_e}\right)\varphi_e \\ \text{subject to} \quad & \varphi_v \in \Phi_v,\ x_v \in X_v, \qquad \forall v \in V \\ & y_e = \varphi_e x_u,\ z_e = \varphi_e x_v, \qquad \forall e = (u,v) \in E \end{aligned}$$

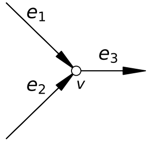
- Convex objective (perspective function of ℓ_e)
- Nonconvex constraints (bilinear)



[Combettes, '16]

Step 2: Compacting the notation

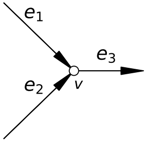
- Each edge has two "spatial" variables y_e, z_e (origin/destination)
- Bilinear constraints $y_e = \varphi_e x_u$ and $z_e = \varphi_e x_v$ for all edges $e = (u, v)$
- At each node, organize variables in matrices



$$M_v = \begin{bmatrix} z_{e_1} & z_{e_2} & y_{e_3} \end{bmatrix} = \begin{bmatrix} \varphi_{e_1} x_v & \varphi_{e_2} x_v & \varphi_{e_3} x_v \end{bmatrix} = x_v \varphi_v^\top$$

Step 2: Compacting the notation

- Each edge has two "spatial" variables y_e, z_e (origin/destination)
- Bilinear constraints $y_e = \varphi_e x_u$ and $z_e = \varphi_e x_v$ for all edges $e = (u, v)$
- At each node, organize variables in matrices

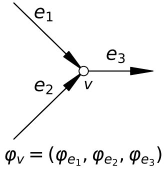


$$M_v = \begin{bmatrix} z_{e_1} & z_{e_2} & y_{e_3} \end{bmatrix} = \begin{bmatrix} \varphi_{e_1} x_v & \varphi_{e_2} x_v & \varphi_{e_3} x_v \end{bmatrix} = x_v \varphi_v^\top$$

- Define $\Omega_v = \left\{ (\varphi, x, M) : \varphi \in \Phi_v,\ x \in X_v,\ M = x\varphi^\top \right\}$

Step 2: Compacting the notation

- Each edge has two "spatial" variables y_e, z_e (origin/destination)
- Bilinear constraints $y_e = \varphi_e x_u$ and $z_e = \varphi_e x_v$ for all edges $e = (u, v)$
- At each node, organize variables in matrices



$$M_v = \begin{bmatrix} z_{e_1} & z_{e_2} & y_{e_3} \end{bmatrix} = \begin{bmatrix} \varphi_{e_1} x_v & \varphi_{e_2} x_v & \varphi_{e_3} x_v \end{bmatrix} = x_v \varphi_v^\top$$

- Define $\Omega_v = \left\{ (\varphi, x, M) : \varphi \in \Phi_v,\ x \in X_v,\ M = x\varphi^\top \right\}$

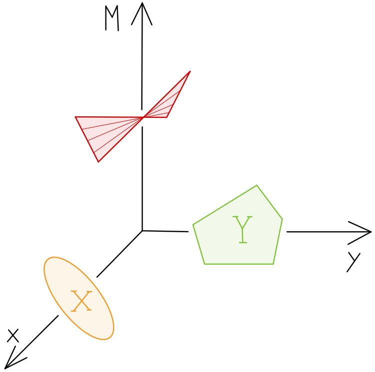
The bilinear program

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} \ell_e \left(\frac{y_e}{\varphi_e}, \frac{z_e}{\varphi_e} \right) \varphi_e \\ \text{subject to} \quad & (\varphi_v, x_v, M_v) \in \Omega_v, \qquad \forall v \in V \end{aligned}$$

Set-based convex relaxation of the bilinearities (I)

Goal: find (good) convex outer approximation Ω' of

$$\Omega = \left\{(x, y, M) : x \in X,\ y \in Y,\ M = xy^\top\right\}$$



Set-based convex relaxation of the bilinearities (I)

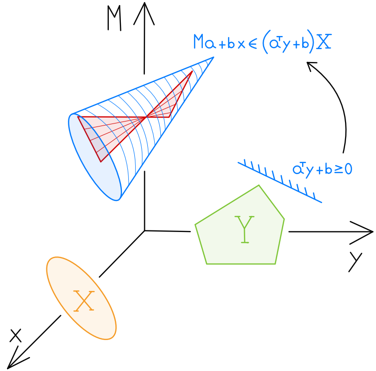
Goal: find (good) convex outer approximation Ω' of

$$\Omega = \left\{ (x, y, M) : x \in X,\ y \in Y,\ M = xy^\top \right\}$$

Main lemma (lifting valid inequalities)

- Assume $a^\top y + b \geq 0$ for all $y \in Y$
- All points $(x, y, M) \in \Omega$ verify the constraint

$$Ma + bx \in (a^\top y + b)X$$



Set-based convex relaxation of the bilinearities (I)

Goal: find (good) convex outer approximation Ω' of

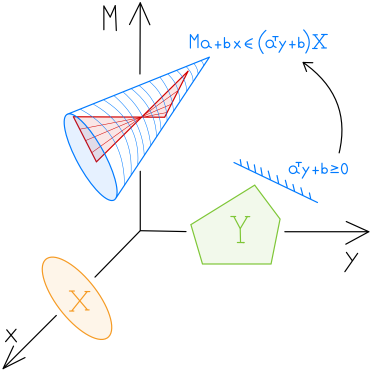
$$\Omega = \left\{(x, y, M) : x \in X,\ y \in Y,\ M = xy^\top\right\}$$

Main lemma (lifting valid inequalities)

- Assume $a^\top y + b \geq 0$ for all $y \in Y$
- All points $(x, y, M) \in \Omega$ verify the constraint

$$Ma + bx \in (a^\top y + b)X$$

- Easy: $x \in X \quad \Rightarrow \quad x(y^T a + b) \in (y^T a + b)X$



Set-based convex relaxation of the bilinearities (I)

Goal: find (good) convex outer approximation Ω' of

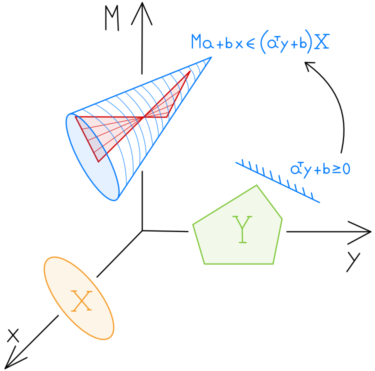
$$\Omega = \left\{(x, y, M) : x \in X,\ y \in Y,\ M = xy^\top\right\}$$

Main lemma (lifting valid inequalities)

- Assume $a^\top y + b \geq 0$ for all $y \in Y$
- All points $(x, y, M) \in \Omega$ verify the constraint

$$Ma + bx \in (a^\top y + b)X$$

- Easy: $x \in X \quad \Rightarrow \quad x(y^T a + b) \in (y^T a + b)X$
- Constraint is convex in (x, y, M) !



Set-based convex relaxation of the bilinearities (I)

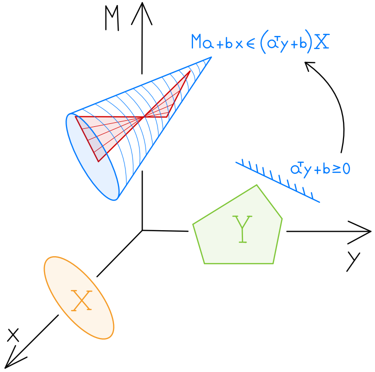
Goal: find (good) convex outer approximation Ω' of

$$\Omega = \left\{(x, y, M) : x \in X,\ y \in Y,\ M = xy^\top\right\}$$

Main lemma (lifting valid inequalities)

- Assume $a^\top y + b \geq 0$ for all $y \in Y$
- All points $(x, y, M) \in \Omega$ verify the constraint

$$Ma + bx \in (a^\top y + b)X$$



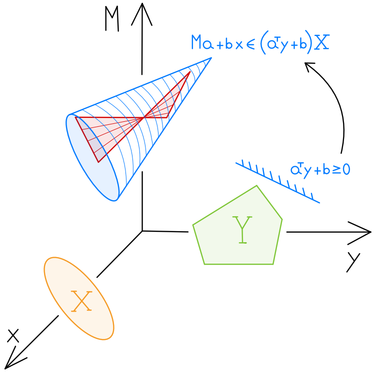
- Easy: $x \in X \quad \Rightarrow \quad x(y^T a + b) \in (y^T a + b)X$
- Constraint is convex in (x, y, M) !
- Set-based relaxation, don't care about *description* of X (cf. Lovasz-Schrijver vs. RLT, also OOP-friendly)

Set-based convex relaxation of the bilinearities (II)

Goal: Find (good) convex outer approximation Ω' of

$$\Omega = \left\{(x, y, M) : x \in X,\ y \in Y,\ M = xy^\top\right\}$$

For simplicity, let Y be polyhedral (e.g., flow polytope).



Set-based convex relaxation of the bilinearities (II)

Goal: Find (good) convex outer approximation Ω' of

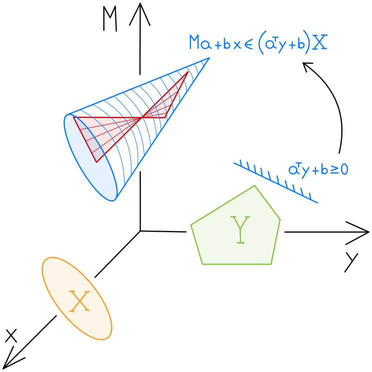
$$\Omega = \left\{(x, y, M) : x \in X,\ y \in Y,\ M = xy^\top\right\}$$

For simplicity, let Y be polyhedral (e.g., flow polytope).

Main lemma (lifting valid inequalities)

- Assume $a^\top y + b \geq 0$ for all $y \in Y$
- All points $(x, y, M) \in \Omega$ verify

$$Ma + bx \in (a^\top y + b)X$$



Set-based convex relaxation of the bilinearities (II)

Goal: Find (good) convex outer approximation Ω' of

$$\Omega = \left\{(x, y, M) : x \in X,\ y \in Y,\ M = xy^\top\right\}$$

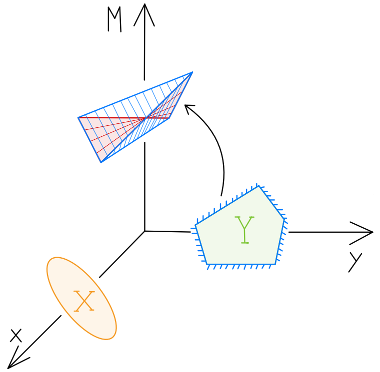
For simplicity, let Y be polyhedral (e.g., flow polytope).

Main lemma (lifting valid inequalities)

- Assume $a^\top y + b \geq 0$ for all $y \in Y$
- All points $(x, y, M) \in \Omega$ verify

$$Ma + bx \in (a^\top y + b)X$$

- Apply lemma to each facet of Y to obtain $\Omega' \supseteq \Omega$

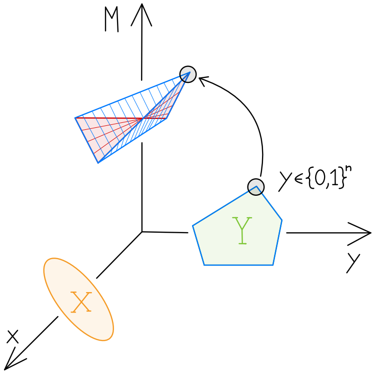


Set-based convex relaxation of the bilinearities (II)

Goal: Find (good) convex outer approximation Ω' of

$$\Omega = \left\{ (x, y, M) : x \in X,\ y \in Y,\ M = xy^\top \right\}$$

For simplicity, let Y be polyhedral (e.g., flow polytope).



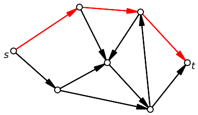
Main lemma (lifting valid inequalities)

- Assume $a^\top y + b \geq 0$ for all $y \in Y$
- All points $(x, y, M) \in \Omega$ verify

$$Ma + bx \in (a^\top y + b)X$$

- Apply lemma to each facet of Y to obtain $\Omega' \supseteq \Omega$
- (persistence) $\Omega' = \Omega$ when restricting $y \in \{0, 1\}^n$

Shortest path: standard vs GCS



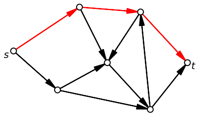
Problem statement

$$\min \sum_{e \in E_\pi} c_e \quad \text{s.t.} \quad \pi \in \Pi$$

(Mixed Integer) Convex program

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \varphi_e \\ \text{s.t.} \quad & \sum_{e \in I_v} \varphi_e - \sum_{e \in O_v} \varphi_e = \delta_{sv}, \quad \forall v \in V - \{t\} \\ & \varphi_e \geq 0, \quad \forall e \in E \qquad \varphi_e \in \{0, 1\} \end{aligned}$$

Shortest path: standard vs GCS

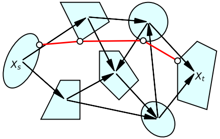


Problem statement

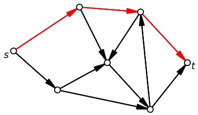
$$\min \sum_{e \in E_\pi} c_e \qquad \text{s.t.} \quad \pi \in \Pi$$

(Mixed Integer) Convex program

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \varphi_e \\ \text{s.t.} \quad & \sum_{e \in I_v} \varphi_e - \sum_{e \in O_v} \varphi_e = \delta_{sv}, \quad \forall v \in V - \{t\} \\ & \varphi_e \geq 0, \quad \forall e \in E \qquad \varphi_e \in \{0, 1\} \end{aligned}$$



Shortest path: standard vs GCS

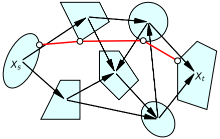


Problem statement

$$\min \quad \sum_{e \in E_\pi} c_e$$
$$\text{s.t.} \quad \pi \in \Pi$$

(Mixed Integer) Convex program

$$\min \quad \sum_{e \in E} c_e \varphi_e$$
$$\text{s.t.} \quad \sum_{e \in I_v} \varphi_e - \sum_{e \in O_v} \varphi_e = \delta_{sv}, \quad \forall v \in V - \{t\}$$
$$\varphi_e \geq 0, \quad \forall e \in E \qquad \varphi_e \in \{0, 1\}$$



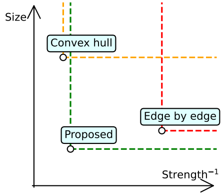
$$\min \quad \sum_{e \in E_\pi} \ell_e(x_u, x_v)$$
$$\text{s.t.} \quad \pi \in \Pi$$
$$x_v \in X_v, \quad \forall v \in \pi$$

$$\min \quad \sum_{e \in E} \ell_e\left(\frac{y_e}{\varphi_e}, \frac{z_e}{\varphi_e}\right)\varphi_e$$
$$\text{s.t.} \quad \sum_{e \in I_v} \begin{bmatrix} z_e \\ \varphi_e \end{bmatrix} - \sum_{e \in O_v} \begin{bmatrix} y_e \\ \varphi_e \end{bmatrix} = \delta_{sv} \begin{bmatrix} x_v \\ 1 \end{bmatrix}, \quad \forall v \in V - \{t\}$$
$$y_e \in \varphi_e X_u, \; z_e \in \varphi_e X_v, \quad \forall e = (u, v) \in E$$
$$\varphi_e \geq 0, \quad \forall e \in E \qquad \varphi_e \in \{0, 1\}$$

Further comments

Convex relaxation

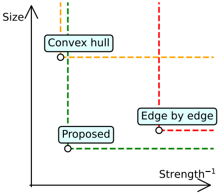
- Applies to sets $\{(\alpha, \beta, M) : \alpha \in A,\ \beta \in B,\ M = xy^\top\}$
- Very tight in practice
  - Exact when sets X_v are singletons
- Compact: $O((|V| + |E|)\dim(X_v))$ variables and constraints
- Typically, linear or second-order-cone programs



Further comments

Convex relaxation

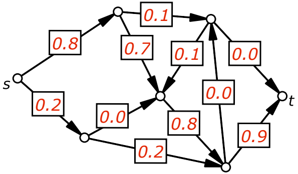
- Applies to sets $\{(\alpha, \beta, M) : \alpha \in A,\ \beta \in B,\ M = xy^\top\}$
- Very tight in practice
  - Exact when sets X_v are singletons
- Compact: $O((|V| + |E|)\dim(X_v))$ variables and constraints
- Typically, linear or second-order-cone programs



What if (after solving) flows are not 0/1?

A. Rounding the solution of the convex relaxation

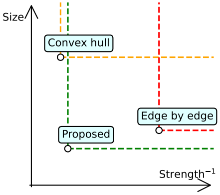
- Flows $\varphi_e \in [0, 1]$ are interpretable as probabilities
- Automatically provides optimality bounds



Further comments

Convex relaxation

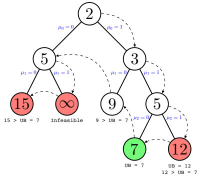
- Applies to sets $\{(\alpha, \beta, M) : \alpha \in A,\ \beta \in B,\ M = xy^\top\}$
- Very tight in practice
  - Exact when sets X_v are singletons
- Compact: $O((|V| + |E|)\dim(X_v))$ variables and constraints
- Typically, linear or second-order-cone programs



What if (after solving) flows are not 0/1?

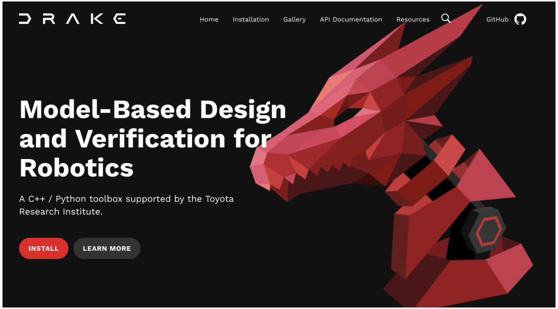
A. Rounding the solution of the convex relaxation
   - Flows $\varphi_e \in [0, 1]$ are interpretable as probabilities
   - Automatically provides optimality bounds

B. Branch and bound



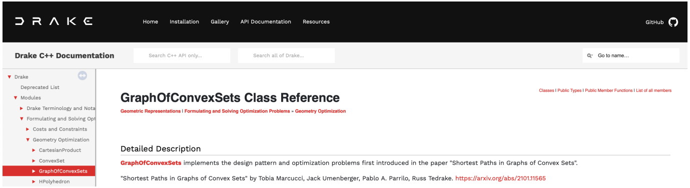
Mature software implementation in Drake

- Implemented in robotics software Drake (TRI), C++/Python
- Off-the-shelf solvers (Gurobi/MOSEK/SCS/CSDP/ SNOPT/IPOPT/NLOPT)
- Customized solvers/algorithms (experimental)

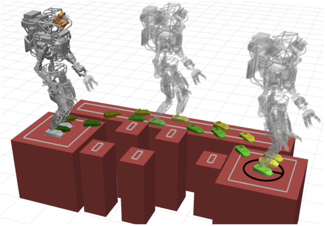


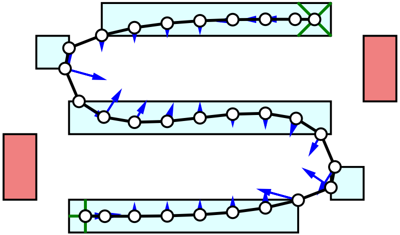
Give it a try: `pip install drake`

TOYOTA RESEARCH INSTITUTE

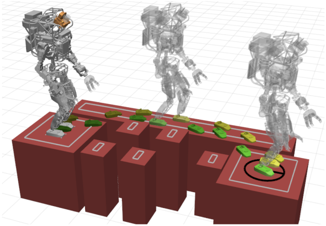


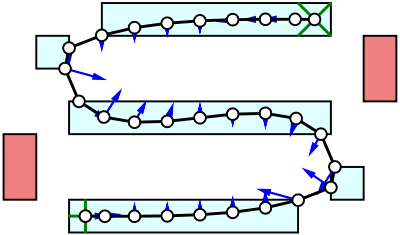
Optimal control of a hybrid system



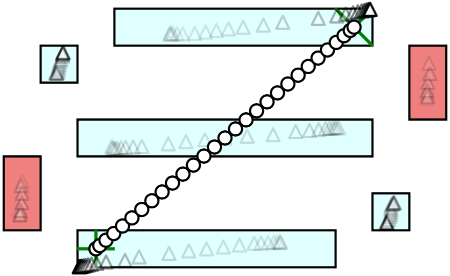


Optimal control of a hybrid system



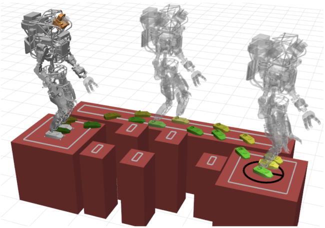


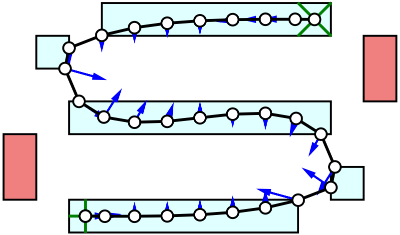
[Moehle and Boyd '15], [Marcucci and Tedrake '19]



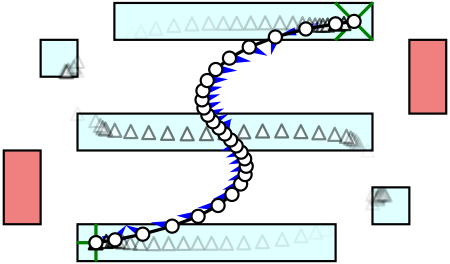
- $\dfrac{\text{Cost relaxation}}{\text{Cost MICP}} = 7\%$
- Branch-and-bound time 218s

Optimal control of a hybrid system



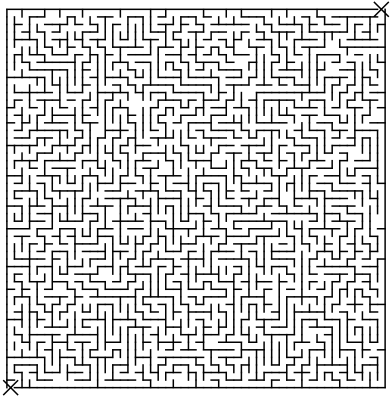


Shortest-path formulation



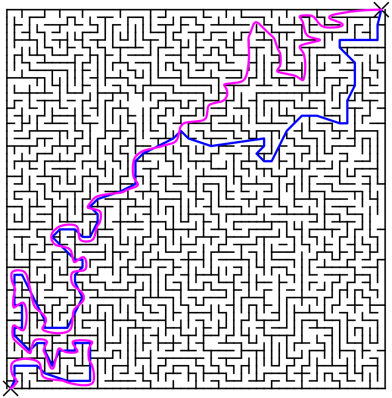
- $\dfrac{\text{Cost relaxation}}{\text{Cost MICP}} = 80\%$
- Branch-and-bound time 1.3s

Motion planning in a maze



- Easy to find a path via discrete graph search
- If we have differential costs and constraints?
  - Local optimization: hopeless
  - Sampling based: inefficient and non-differentiable
  - Prev. mixed-integer: $(\#\text{ cells})^2 \approx 6 \cdot 10^6$ binaries

Motion planning in a maze



- Easy to find a path via discrete graph search
- If we have differential costs and constraints?
  - Local optimization: hopeless
  - Sampling based: inefficient and non-differentiable
  - Prev. mixed-integer: $(\# \text{ cells})^2 \approx 6 \cdot 10^6$ binaries

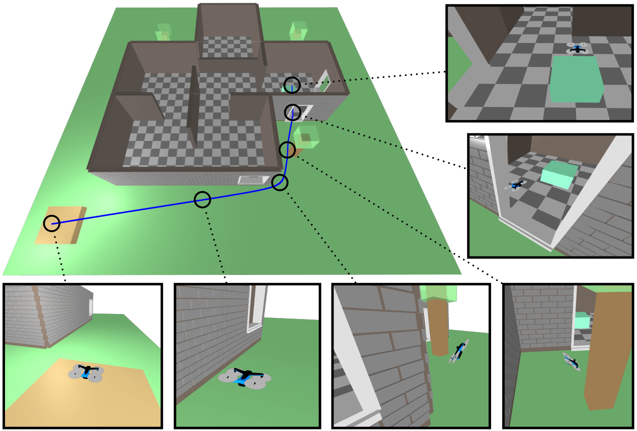
Graphs of convex sets

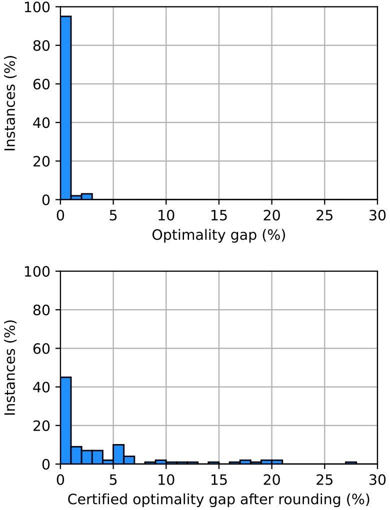
- Minimum distance
- Minimum time
  - With velocity limits and acceleration penalty
- Convex relaxation is exact in both cases!
- Only $O(\# \text{ cells})$ flow variables

Quadrotor flying around obstacles

- Exact decomposition of free space in convex sets
- Planning in (x, y, z) + differential flatness
- Penalties on length, velocity, acceleration, and duration

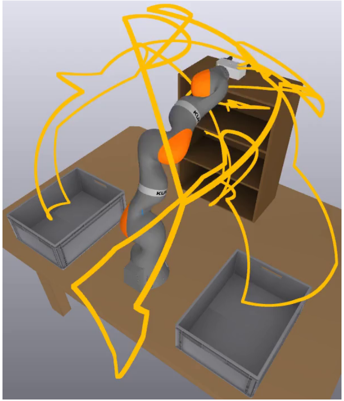
Convex relaxation + randomized rounding



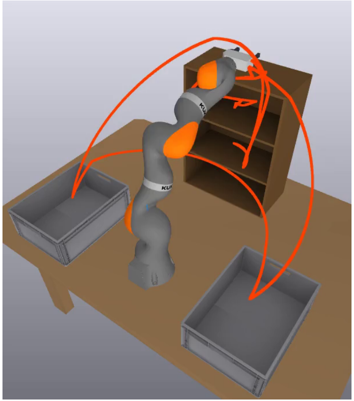


Comparison with Probabilistic RoadMap (PRM)

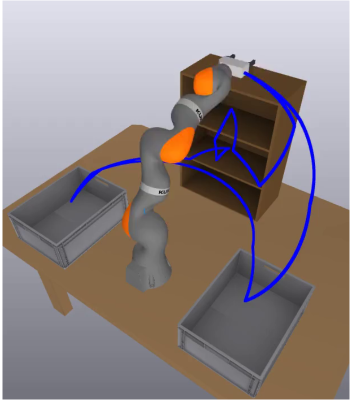
PRM



PRM with shortcuts



Graph of Convex Sets (GCS)



- Convex relaxation + rounding
- [Amice et al., '22] for the decomposition of configuration space

Comparison with Probabilistic RoadMap (PRM)

PRM	PRM with shortcuts	Graph of Convex Sets (GCS)

- Convex relaxation + rounding
- [Amice et al., '22] for the decomposition of configuration space

Planning in 14 dimensions

- Collision-free motion planning in 14 dimensions using convex optimization
- PRM hardly scales beyond 7/8 dimensions

Preliminary hardware results

Motion generated via a single convex optimization!

Wrapping up

Shortest-path problem in graphs of convex sets

- Exciting new optimization framework, flexible and powerful
- Efficiently solvable in practice
  - Tight convex relaxation + rounding
  - Strong mixed-integer convex formulation + branch and bound

Current and future directions

- Customized ADMM solver on GPU (eventually a standalone toolbox)
- Alternative algorithmic approaches? Scale to huge graphs?
- Other combinatorial problems in graphs of convex sets (TSP, MSTP)
- Extensions/applications: underactuated dynamics, temporal logic, stochastics, SLAM...

Wrapping up

Shortest-path problem in graphs of convex sets

- Exciting new optimization framework, flexible and powerful
- Efficiently solvable in practice
  - Tight convex relaxation + rounding
  - Strong mixed-integer convex formulation + branch and bound

Current and future directions

- Customized ADMM solver on GPU (eventually a standalone toolbox)
- Alternative algorithmic approaches? Scale to huge graphs?
- Other combinatorial problems in graphs of convex sets (TSP, MSTP)
- Extensions/applications: underactuated dynamics, temporal logic, stochastics, SLAM...
- Your new algorithms?

Thanks for your attention!

Thanks for your attention!

Questions?