Factored Task and Motion Planning with Combined Optimization, Sampling and Learning

Joaquim Ortiz-Haro
PhD Defense
TU Berlin
29 January 2024

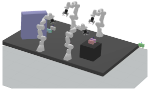
Doctoral Committee
- Reviewer: Tomás Lozano-Pérez (MIT)
- Reviewer: Georg Martius (Uni Tübingen)
- Reviewer and PhD Advisor: Marc Toussaint (TU Berlin)
- Chair: Marc Alexa (TU Berlin)

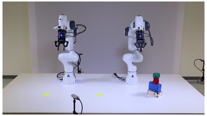
Presentation Overview

Introduction: Task and Motion Planning

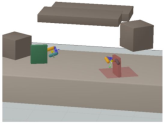
Factored Structure of Task and Motion Planning

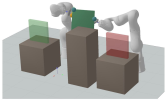
Part I Integrated Planning and Optimization for Task and Motion Planning



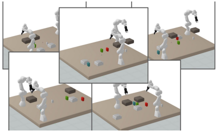


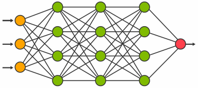
Part II Meta-Solvers: Adaptive Combination of Sampling and Optimization Methods





Part III Accelerated Task and Motion Planning with Learning Methods





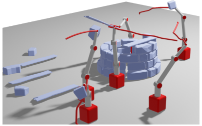
Conclusion and Future Work

Autonomy of Robotic Systems

Robots excel at performing repetitive tasks,
e.g., in car factories.
- Optimal Control (following a reference trajectory)
- Motion Planning (creating a collision-free path).

[1]

But future robotic applications (e.g., in construction, elderly care, home assistance...) will require long-term planning of physical interactions with the environment.



[2]

[3]

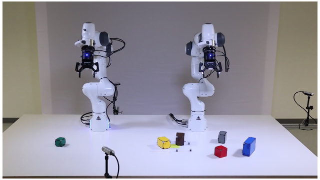
[4]

[5]

Task and Motion Planning

Task and Motion Planning (TAMP) in Robotics.

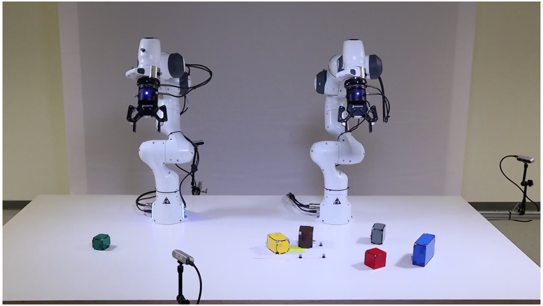
Initial state

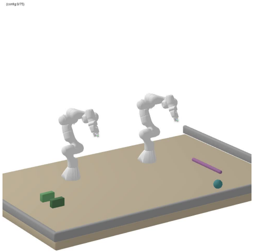


Symbolic goal

`tower blue-gray-red-green`
`in the center of the table`

Assumption: we have a good model of the robot and the environment (e.g., the shape of the objects, where they are ...).

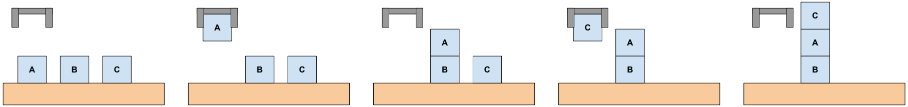




Understanding TAMP in 120 Seconds. Two levels of abstraction

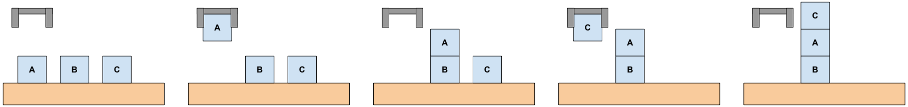
Goal: e.g, build a tower with blocks (requires long-term planning of physical interactions).
(High-level Task Planning): What to do? – e.g., pick the red block with the left robot.
Discrete planning problem (PDDL, STRIPS).

A* with Heuristics This is only a simplification! No continuous information.

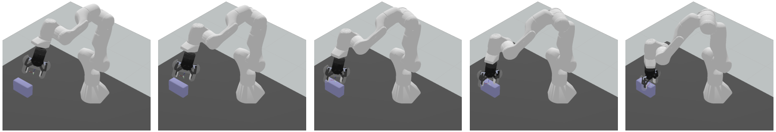


Understanding TAMP in 120 Seconds. Two levels of abstraction

Goal: e.g, build a tower with blocks (requires long-term planning of physical interactions).
(High-level Task Planning): What to do? – e.g., pick the red block with the left robot. Discrete planning problem (PDDL, STRIPS).
A* with Heuristics This is only a simplification! No continuous information.



(Low-level Motion Planning) How to do? Collision-free trajectory, stable grasps, pushing interactions, continuous space. The trajectory must fulfill physics constraints.
Trajectory Optimization and/or Motion Planning. Computationally expensive.



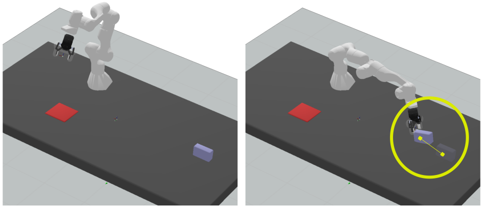
Understanding TAMP in 120 Seconds.

Strong dependencies between task planning and motion planning.

1 - The motion planning problem (cost, collision and constraints) depend on the task plan.

2 - Often task plans fail at the motion level.

Example 1. Task Plan: *Pick object*
– but the object is too far!



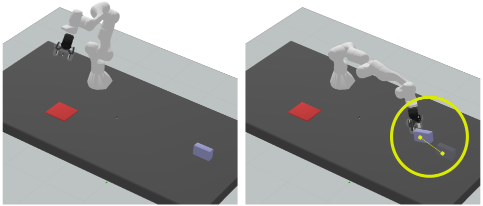
Understanding TAMP in 120 Seconds.

Strong dependencies between task planning and motion planning.

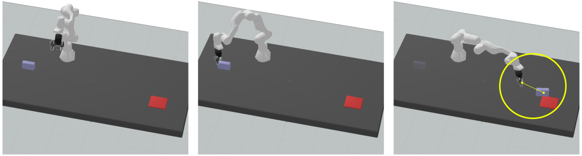
1 - The motion planning problem (cost, collision and constraints) depend on the task plan.

2 - Often task plans fail at the motion level.

Example 1. Task Plan: *Pick object*
– but the object is too far!



Example 2. Task Plan: *Pick object and place object on the table*
– but the table is too far!



Related Work – How Can We Solve TAMP?

How to combine and integrate discrete task planning and continuous motion planning (which tools)?

Related Work – How Can We Solve TAMP?

How to combine and integrate discrete task planning and continuous motion planning (which tools)?

Sample-Based Approaches to TAMP : Incrementally discretize the continuous space. Tools from motion planning: constrained sampling and sample-based motion planning (RRT, PRM). (Garrett et al., 2020; Srivastava et al., 2014; Dantam et al., 2016). Individual/constrained sampling is inefficient if there are long-term dependencies.

Related Work – How Can We Solve TAMP?

How to combine and integrate discrete task planning and continuous motion planning (which tools)?

Sample-Based Approaches to TAMP : Incrementally discretize the continuous space. Tools from motion planning: constrained sampling and sample-based motion planning (RRT, PRM). (Garrett et al., 2020; Srivastava et al., 2014; Dantam et al., 2016). Individual/constrained sampling is inefficient if there are long-term dependencies.

Optimization-Based Approaches to TAMP : Compute a motion that fulfills a high-level plan with optimization methods. (Toussaint et al., 2018). Good joint reasoning, but difficult to scale to longer tasks.

Related Work – How Can We Solve TAMP?

How to combine and integrate discrete task planning and continuous motion planning (which tools)?

Sample-Based Approaches to TAMP : Incrementally discretize the continuous space. Tools from motion planning: constrained sampling and sample-based motion planning (RRT, PRM). (Garrett et al., 2020; Srivastava et al., 2014; Dantam et al., 2016). Individual/constrained sampling is inefficient if there are long-term dependencies.

Optimization-Based Approaches to TAMP : Compute a motion that fulfills a high-level plan with optimization methods. (Toussaint et al., 2018). Good joint reasoning, but difficult to scale to longer tasks.

The TAMP problem appears under other names: multi-modal planning, manipulation planning, hybrid planning, contact planning, AI planning with numerical variables ...

Research Statement

Improve general-purpose task and motion planning by better leveraging the problem structure and a more effective combination of algorithmic and planning tools.

Research Statement

Improve general-purpose task and motion planning by better leveraging the problem structure and a more effective combination of algorithmic and planning tools.

General-Purpose TAMP

- Pick, place, and push
- Handover and assembly
- Tool utilization
- Multi-robot coordination
- Mobile and fixed robots

Problem Structure

Temporal dimenson, multiple objects, and multiple robots.

Algorithmic and Planning Tools

Trajectory optimization, constrained sampling, discrete planning, and learning.

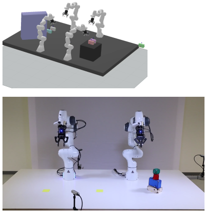
Presentation Overview

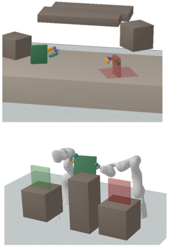
Factored Structure of Task and Motion Planning (Ch. 3)

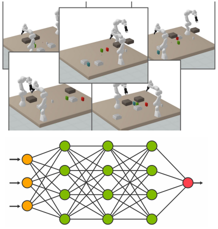
Part I Integrated Planning and Optimization for Task and Motion Planning

Part II Meta-Solvers: Adaptive Combination of Sampling and Optimization Methods

Part III Accelerated Task and Motion Planning with Learning Methods







Factored Structure of Task and Motion Planning

Task plan

$\downarrow$

Motion planning problem

(Unfactored)
Nonlinear program

$$\min f(x,\ \text{Task plan}),$$

$$\text{s.t. } h(x,\ \text{Task plan}) = 0,$$

$$g(x,\ \text{Task plan}) \leq 0.$$

$$x = [b_0, q_0, b_1, q_1, \tau_1^b, \tau_2^b, ...]$$

h and g are vector-valued constraint functions.

Factored Structure of Task and Motion Planning

Task plan
↓
Motion planning problem

(Unfactored)
Nonlinear program

$\min f(x,\ \text{Task plan}),$

$\text{s.t. } h(x,\ \text{Task plan}) = 0,$

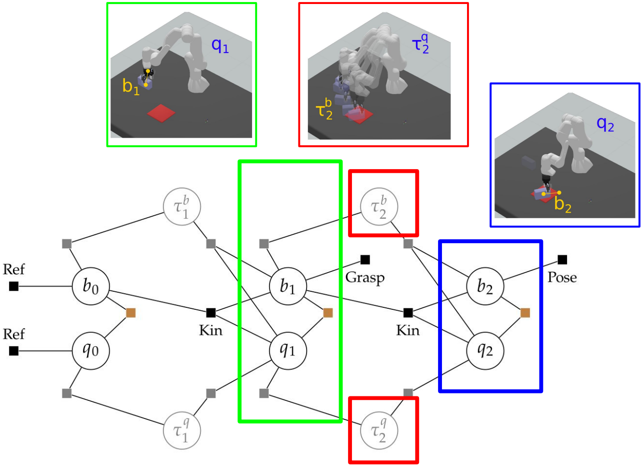
$g(x,\ \text{Task plan}) \leq 0.$

$x = [b_0, q_0, b_1, q_1, \tau_1^b, \tau_2^b, \ldots]$

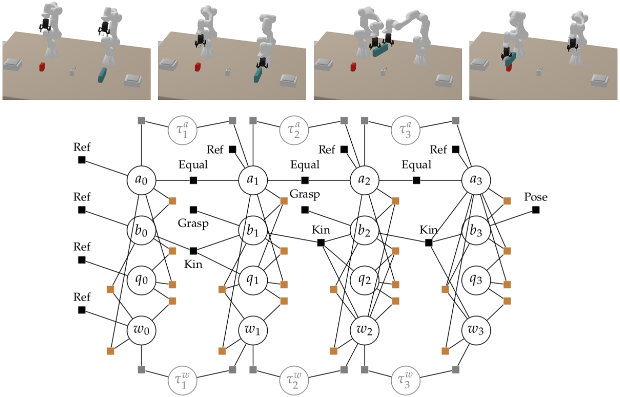
h and g are vector-valued constraint functions.

Factored nonlinear program

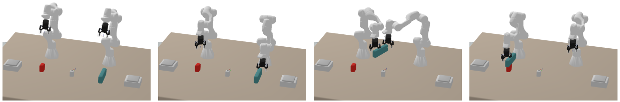
Task plan: *Pick Object, Place Object*

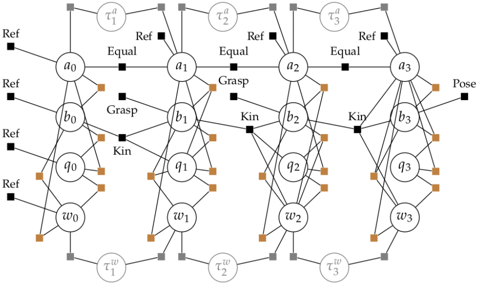


Different high-level task plans imply different Factored-NLPs!
– But they share the same small building blocks.



Different high-level task plans imply different Factored-NLPs!
– But they share the same small building blocks.



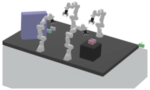


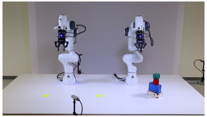
3 key properties: Temporal structure, sparse factorization, repeatable local structure.

Equivalent factored representations have been used in recent Sample-Based TAMP solvers Garrett et al. (2018); Lagriffoul et al. (2014). We contribute a new formulation and novel applications in planning and learning.

Presentation Overview

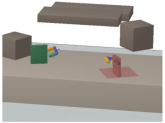
Part I Integrated Planning and Optimization for Task and Motion Planning

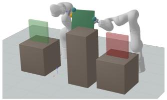




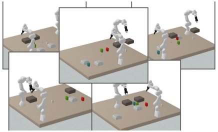
(Ch. 4 ICAPS 2021)
(Ch. 5 RAL 2022)

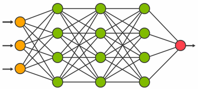
Part II Meta-Solvers: Adaptive Combination of Sampling and Optimization Methods



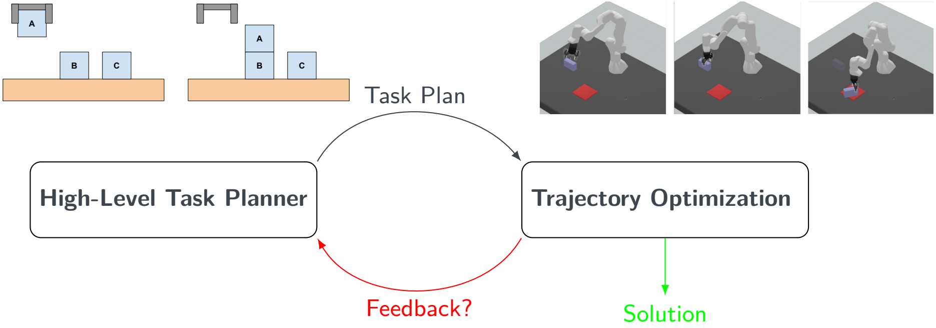


Part III Accelerated Task and Motion Planning with Learning Methods





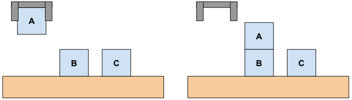
Part I. Integrated Planning and Optimization for Task and Motion Planning

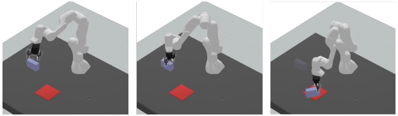


Feedback when the plan fails is important!

- No feedback Failure.
- Feedback = Task Plan Inefficient. E.g., with 5 objects and 2 robots, there are approximately $(2 \cdot 5)^{10}$ plans of length 10.

Part I – Ch. 5: Conflict-Based Search in Factored Logic Geometric Programs



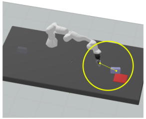


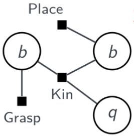
Task Plan

High-Level Task Planner → **Trajectory Optimization**

Subsets of nonlinear constraints

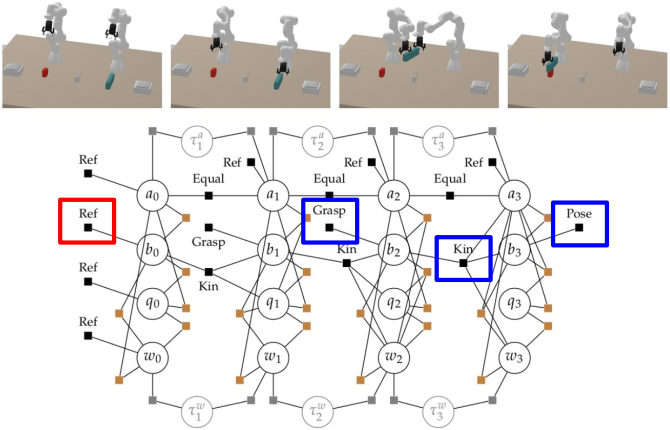
Solution





Ortiz-Haro, J., Karpas, E., Katz, M., and Toussaint, M. (2022). A Conflict-Driven Interface Between Symbolic Planning and Nonlinear Constraint Solving. IEEE Robotics and Automation Letters.

Bidirectional Factored interface between ‘predicates‘ (partial states) in the task plan and ‘constraints‘ in the trajectory optimization problem.



Object A is on the initial position

Robot is holding object B $\rightarrow$ Object B is on top of A

Two technical contributions

1 - How to find a minimal subset of infeasible constraints?

2 - How to reformulate the planning problem to block this conflict?

Two technical contributions

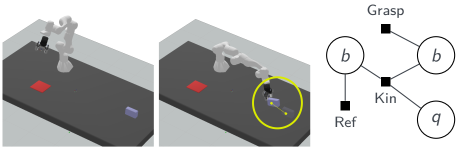
1 - How to find a minimal subset of infeasible constraints?

2 - How to reformulate the planning problem to block this conflict?

Example of infeasible nonlinear constraints

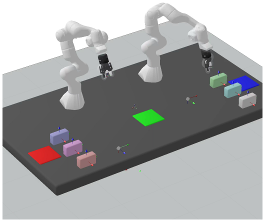
Pick object

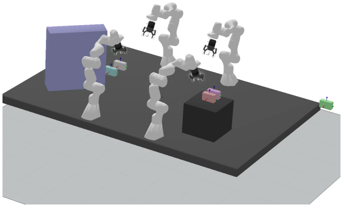
– but the object is too far!

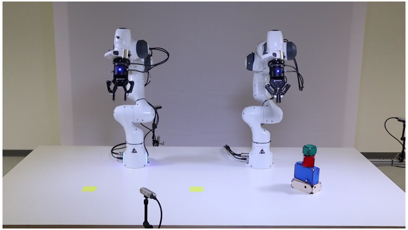


This is only one example! We can discover any conflict, potentially involving multiple motion phases, robots, objects, collisions ...

Results







○ ○ ○

- Complete and general (assuming completeness of the nonlinear solver!)
- Planning time: 2-30 seconds.

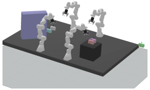
Benchmark

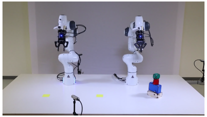
Previous optimization-based solvers (e.g., MBTS):	2 robots, 4 objects, 8 actions.
Ours	4 robots, 8 objects, 24 actions.

Exponential complexity! Adding 1 object makes the problem x2 harder.

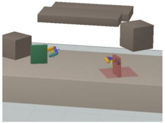
Presentation Overview

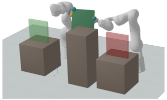
Part I Integrated Planning and Optimization for Task and Motion Planning





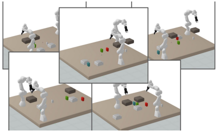
Part II Meta-Solvers: Adaptive Combination of Sampling and Optimization Methods

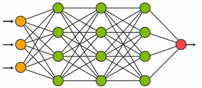




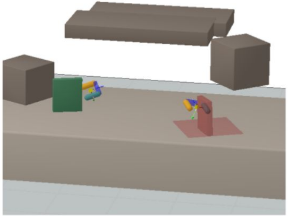
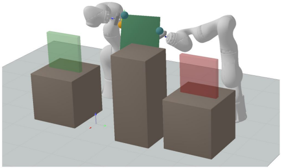
Ch. 6 - ICRA 2021
Ch. 7 - Preprint

Part III Accelerated Task and Motion Planning with Learning Methods



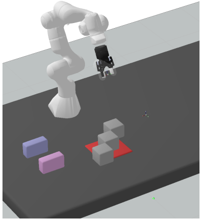


Part II - Meta-Solvers: Adaptive Combination of Sampling and Optimization Methods

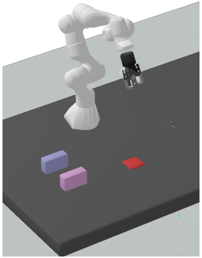
Sampling (decomposition)	**Optimization** (No decomposition)
	
First grasp, then robot, ...	All variables jointly
✓Problem is decomposable	✓Joint dependencies
✗Joint dependencies	✗Infeasible local optima

Part II - Ch. 7: Towards Meta-Solvers for Task and Motion Planning

Symbolic Goal:
"Put the two blocks on the red table"



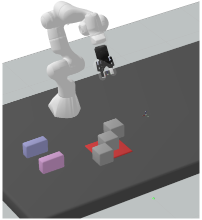
Use sampling better!



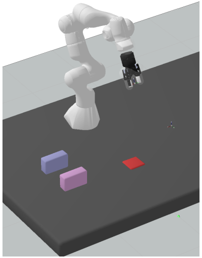
Use optimization better!

Part II - Ch. 7: Towards Meta-Solvers for Task and Motion Planning

Symbolic Goal:
"Put the two blocks on the red table"



Use sampling better!



Use optimization better!

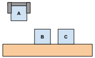
TAMP Solver = Task Plan + Motion Plan
TAMP Meta-Solver = Task Plan + Motion Plan + **Optimization/Sampling Strategy**

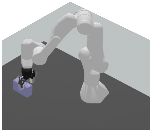
Meta-Solver useful for non-expert users + good performance in any problem.

Ortiz-Haro, J., Erez Karpas and Marc Toussaint. Towards Meta-Solvers for Task and Motion Planning. Preprint. Future submission to ICRA 2025, or ICAPS 2025.

How to design a TAMP Meta-Solver?

Discrete-continuous state in TAMP: $(\mathbf{s}, \mathbf{x})$





Discrete s Continuous x (free or assigned)

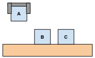
To bridge the gap we need a more general representation: **the computational state.**

Computational State in TAMP: $(\mathbf{s}, \mathbf{x}, \tilde{X}, \Phi)$

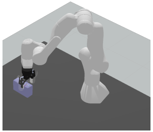
- $s \in \mathcal{S}$ is a discrete state.
- $x \in \mathcal{X}$ is a fixed continuous state.
- $\tilde{X}$ is a set of free continuous states.
- Φ is a set of nonlinear constraints on the free states.

How to design a TAMP Meta-Solver?

Discrete-continuous state
in TAMP: $(\mathbf{s}, \mathbf{x})$



Discrete s



Continuous x (free or assigned)

To bridge the gap we need a more general representation: **the computational state.**

Computational State in TAMP: $(\mathbf{s}, \mathbf{x}, \tilde{X}, \Phi)$

- $s \in \mathcal{S}$ is a discrete state.
- $x \in \mathcal{X}$ is a fixed continuous state.
- $\tilde{X}$ is a set of free continuous states.
- Φ is a set of nonlinear constraints on the free states.

Planning in computational space. Two type of compute actions:

- Compute values for free variables.
- Extend the high-level task plan (e.g, ‘pick object‘) (changes the discrete state and creates more free variables with constraints).

We can recover traditional TAMP solvers as special search algorithms in the space of computational states.

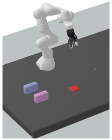
- Optimization-Based TAMP Solver: “MultiBound Tree Search for LGP”
- Sample-Based TAMP Solvers: “PDDLStream”

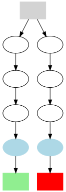
We can recover traditional TAMP solvers as special search algorithms in the space of computational states.

- Optimization-Based TAMP Solver: “MultiBound Tree Search for LGP”
- Sample-Based TAMP Solvers: “PDDLStream”

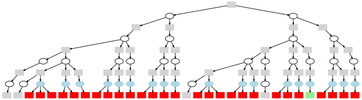
The Meta-solver is an informed search algorithm in the space of computational states.

- Heuristic: discrete task planning.
- Incrementally enumerates the number of times we repeat a numeric expansion.

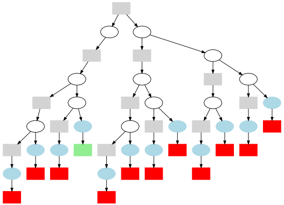




Optimization (Best!)



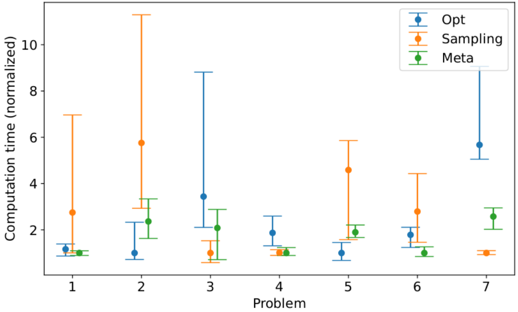
Sampling (Worst!)

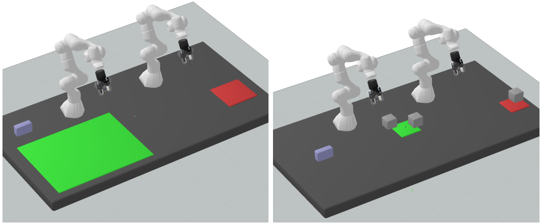


Meta-Solver (Second Best!)

○ With Free Variables ■ Without Free Variables ■ Failed Numeric Expansion ■ Solution

Results





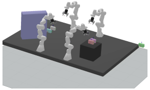
6 7

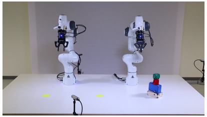
Our “simple” meta-solver already outperforms both Opt./Sample-based TAMP Solvers.

Limitation: the current meta-solver cannot scale to more objects or plans that require a lot of actions!

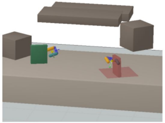
Presentation Overview

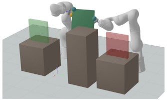
Part I Integrated Planning and Optimization for Task and Motion Planning



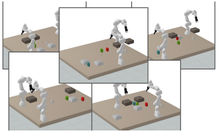


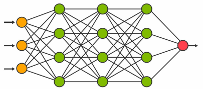
Part II Meta-Solvers: Adaptive Combination of Sampling and Optimization Methods





Part III Accelerated Task and Motion Planning with Learning Methods

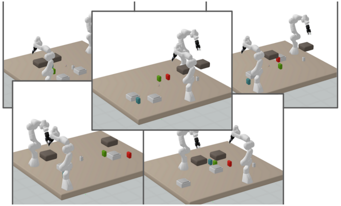




Ch.8 - CoRL 21
Ch.9 - ICRA 23

Part III – Accelerated Task and Motion Planning with Learning Methods

Why do we need data and learning in model-based Task and Motion Planning?

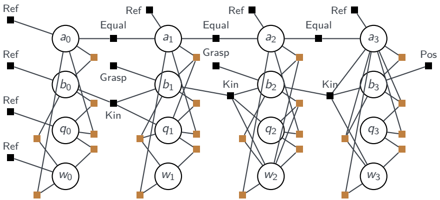


Offline: Generate a dataset of solutions with our solver + Learn weights of a parametric function (neural network). Slow!

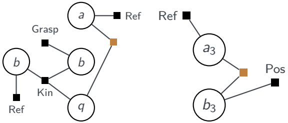
Online: Use the learned function (heuristic, classifier) to accelerate our solver on new problems. Fast!

Part III – Ch. 9: Learning Feasibility of Factored Nonlinear Programs

Input: Factored nonlinear program



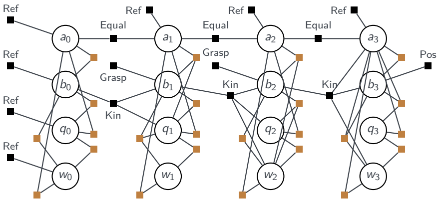
Output: Minimal infeasible subsets of constraints



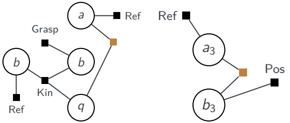
W/o Learning	Factored NLP		Conflict Extraction
Ours	Factored NLP	Graph Neural Network → Small Candidate	Conflict Extraction

Part III – Ch. 9: Learning Feasibility of Factored Nonlinear Programs

Input: Factored nonlinear program



Output: Minimal infeasible subsets of constraints

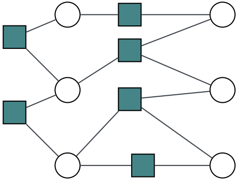


W/o Learning	Factored NLP		Conflict Extraction
Ours	Factored NLP	Graph Neural Network → Small Candidate	Conflict Extraction

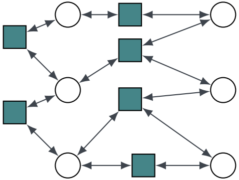
Conflict extraction – remove one constraint at a time and solve the nonlinear program again (linear complexity on the number of constraints).

Learning Feasibility of Factored Nonlinear Programs in Robotic Manipulation Planning J. Ortiz-Haro, J.-S. Ha, D. Driess, E. Karpas, and M. Toussaint. IEEE Int. Conf. on Robotics and Automation (ICRA), 2023.

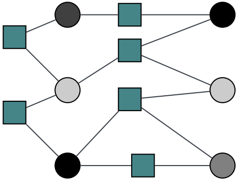
How to predict infeasible subsets of variables and constraints?



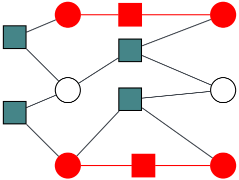
Factored NLP



Neural message passing



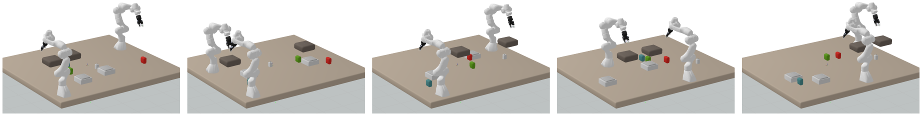
Node scores



Infeasible subgraphs

- Neural message passing
- Node classifier = probability of infeasibility
- Infeasible subgraphs = Filter + connected components analysis

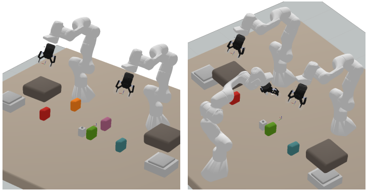
Increase/decrease the threshold to get more/less candidates.

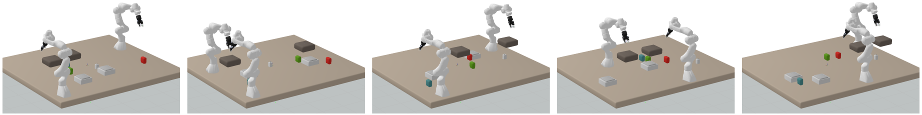


Generalization +action/blocks/robots

(Training dataset: 5000 labelled factored NLPs)

Scene (object, table and robot positions) is encoded locally in the feature vector. **Additional variables and constraints share networks!** Alternative architectures and representations cannot generalize.

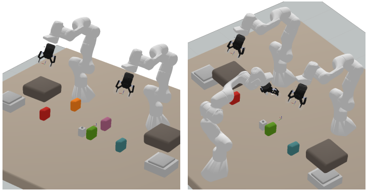




Generalization +action/blocks/robots

(Training dataset: 5000 labelled factored NLPs)

Scene (object, table and robot positions) is encoded locally in the feature vector. **Additional variables and constraints share networks!** Alternative architectures and representations cannot generalize.



4-50x Acceleration in conflict extraction

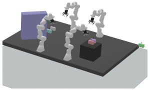
95% Node accuracy

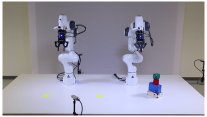
65% Conflicts found, 45% are minimal

Presentation Overview

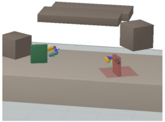
Introduction: Task and Motion Planning

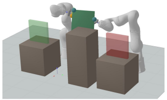
Part I Integrated Planning and Optimization for Task and Motion Planning



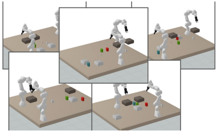


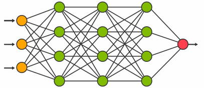
Part II Meta-Solvers: Adaptive Combination of Sampling and Optimization Methods





Part III Accelerated Task and Motion Planning with Learning Methods





Conclusion and Future Work

Conclusion: Summary of Contributions

Factored Nonlinear Program in TAMP – General-purpose, problem-independent, useful representation for both planning and learning. (Ch. 3, also appears in Ch. 5,6,8 and 9).

Part I Integrated Planning and Optimization for Task and Motion Planning

Combine discrete planning with trajectory optimization with a conflict-based bidirectional interface (task plan prefixes or subsets of infeasible constraints).
(Ch. 4, ICAPS 22)
(Ch. 5, RAL 22).

Part II Meta-Solvers: Adaptive Combination of Sampling and Optimization Methods

Neither optimization nor sampling is superior. We need mixed approaches that can adaptively choose compute operations.
(Ch. 6, ICAPS 22)
(Ch. 7, Preprint)

Part III Accelerated Task and Motion Planning with Learning Methods

Acceleration of expensive model-based operations. Different architectures for two different operations.
(Ch. 8, CoRL 21)
(Ch. 9, ICRA 23)

Conclusion: Limitations

Factored Nonlinear Program in TAMP – It requires a custom software implementation. No off-the-shelf simulators or trajectory optimization frameworks.

Part I Integrated Planning and Optimization for Task and Motion Planning

Not complete if the nonlinear solver fails to find a solution (e.g. due to a bad initialization).

Part II Meta-Solvers: Adaptive Combination of Sampling and Optimization Methods

Software complexity and engineering effort. Worse scalability to large TAMP problems.

Part III Accelerated Task and Motion Planning with Learning Methods

Small learning component in a full model-based solver – it requires solvers, models, and data.

Open Challenges and Future Work

- TAMP Benchmarks – Difficult to measure progress. PDDL (discrete planning) and OpenAI Gym (RL) are good inspirations.

Open Challenges and Future Work

- TAMP Benchmarks – Difficult to measure progress. PDDL (discrete planning) and OpenAI Gym (RL) are good inspirations.
- Optimization-Based Solvers in Robotics:
  It works most of the time – this is not enough for real-world applications and broader adoption! Better restart/warm-start strategy.

Open Challenges and Future Work

- TAMP Benchmarks – Difficult to measure progress. PDDL (discrete planning) and OpenAI Gym (RL) are good inspirations.
- Optimization-Based Solvers in Robotics:
  It works most of the time – this is not enough for real-world applications and broader adoption! Better restart/warm-start strategy.
- Learning in TAMP – We want our models to generalize to new problems and new environments. Leverage model-based structure and data of physical interactions for learning universal policies.

Open Challenges and Future Work

- TAMP Benchmarks – Difficult to measure progress. PDDL (discrete planning) and OpenAI Gym (RL) are good inspirations.
- Optimization-Based Solvers in Robotics:
  It works most of the time – this is not enough for real-world applications and broader adoption! Better restart/warm-start strategy.
- Learning in TAMP – We want our models to generalize to new problems and new environments. Leverage model-based structure and data of physical interactions for learning universal policies.
- Perception for TAMP Manipulation and Precise Contact Planning
  Overlooked in this thesis. Robust systems require integrated perception, planning, and control.

  But mastering first model-based TAMP is fundamental! **Long-term planning in continuous spaces is very hard! – Structure, models and planning will help.**

Thanks to

all my collaborators and co-authors!

Learning and Intelligent System Research group at TU-Berlin
Marc Toussaint, Wolfgang Hönig, Ilaria Cicchetti-Nilsson, Jung-Su Ha, Andreas Orthey, Ozgur S. Oguz, Danny Driess, Valentin Noah Hartmann, Svetlana Levit, Akmaral Moldagalieva, Ingmar Schubert, Khaled Wahba, Pia Hanfeld, Cornelius Braun, Hongyou Zhou, Lara Brudermüller.
External Erez Karpas (Technion), Michael Katz (IBM).

Family and Friends

Committee:
Prof. Dr. Marc Toussaint, Prof. Dr. Georg Martius, Prof. Dr. Tomás Lozano-Pérez and Prof. Dr. Marc Alexa.

Thanks for your attention!

References I

Dantam, N. T., Kingston, Z. K., Chaudhuri, S., and Kavraki, L. E. (2016). Incremental task and motion planning: A constraint-based approach. In *Robotics: Science and systems*.

Garrett, C. R., Lozano-Pérez, T., and Kaelbling, L. P. (2018). Sampling-based methods for factored task and motion planning. *The International Journal of Robotics Research*.

Garrett, C. R., Lozano-Pérez, T., and Kaelbling, L. P. (2020). Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of ICAPS*.

Lagriffoul, F., Dimitrov, D., Bidot, J., Saffiotti, A., and Karlsson, L. (2014). Efficiently combining task and motion planning using geometric constraints. *The International Journal of Robotics Research*.

Srivastava, S., Fang, E., Riano, L., Chitnis, R., Russell, S. J., and Abbeel, P. (2014). Combined task and motion planning through an extensible planner-independent interface layer. In *ICRA*.

Toussaint, M., Allen, K. R., Smith, K. A., and Tenenbaum, J. B. (2018). Differentiable physics and stable modes for tool-use and manipulation planning. In *Robotics: Science and Systems XIV RSS*.

Image References

[1] https://www.wevolver.com/article/robots-in-the-manufacturing-industry-types-and-applications
[2] Long-Horizon Multi-Robot Rearrangement Planning for Construction Assembly, Valentin N. Hartmann, Andreas Orthey, Danny Driess, Ozgur S. Oguz, Marc Toussaint. IEEE T-RO 2022.
[3] https://www.newstalk.com/news/robots-are-coming-to-fill-your-dishwasher-correctly-the-rise-of-assistive-ai-1489976
[4] https://www.bobvila.com/slideshow/7-secrets-of-assembling-ikea-furniture-52331
[5] https://www.istockphoto.com Stock-Fotografie-ID:1340509849