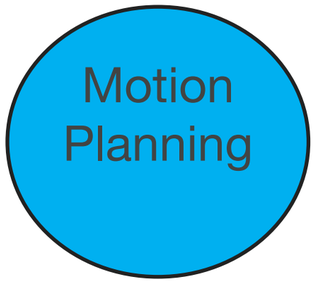
Robotics Solutions Through Scalable Synthesis

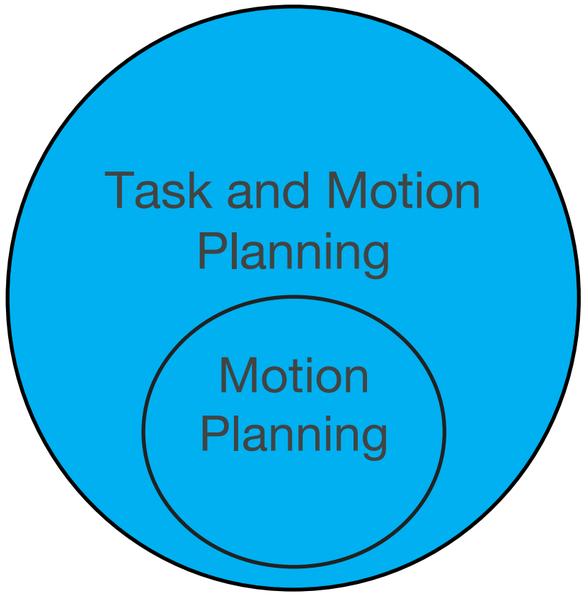
Lydia E. Kavraki
Department of Computer Science
Rice University

From Motion Planning to Task and Motion Planning



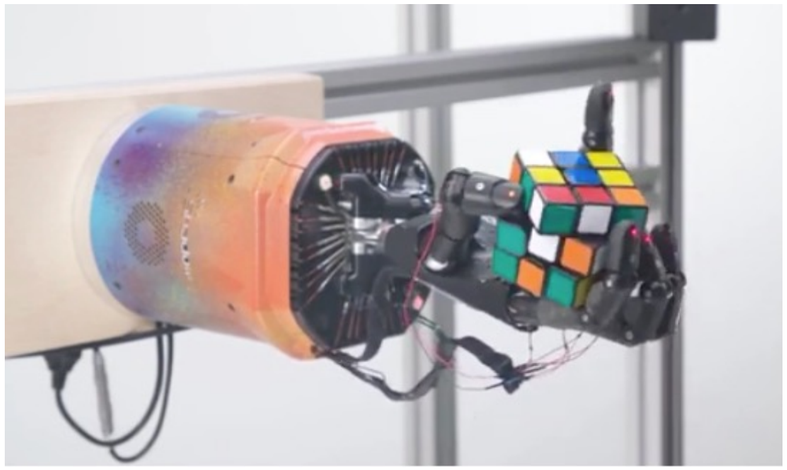
Sampling-based planners

From Motion Planning to Task and Motion Planning



Robots for Specific Tasks

Boston Dynamics



OpenAI

Robots for More General Tasks

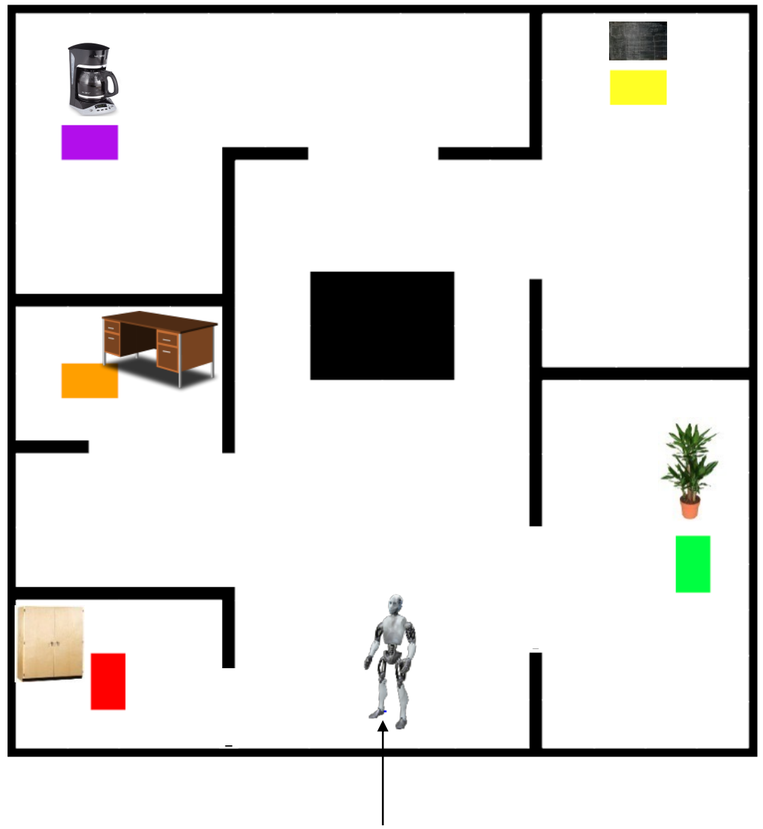
?

How do you Plan a Mission?

Mission:

"Do the following in any order and always avoid the black demo area

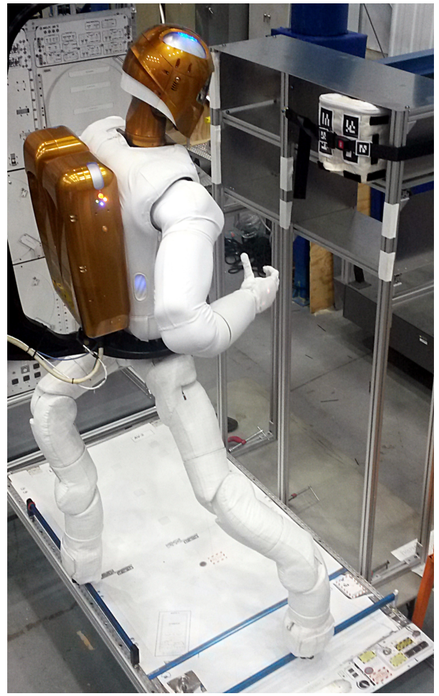
- water the plant
- clean the blackboard
- turn off the coffee maker
- pick up vacuum cleaner from the supply room, then vacuum the conference room."

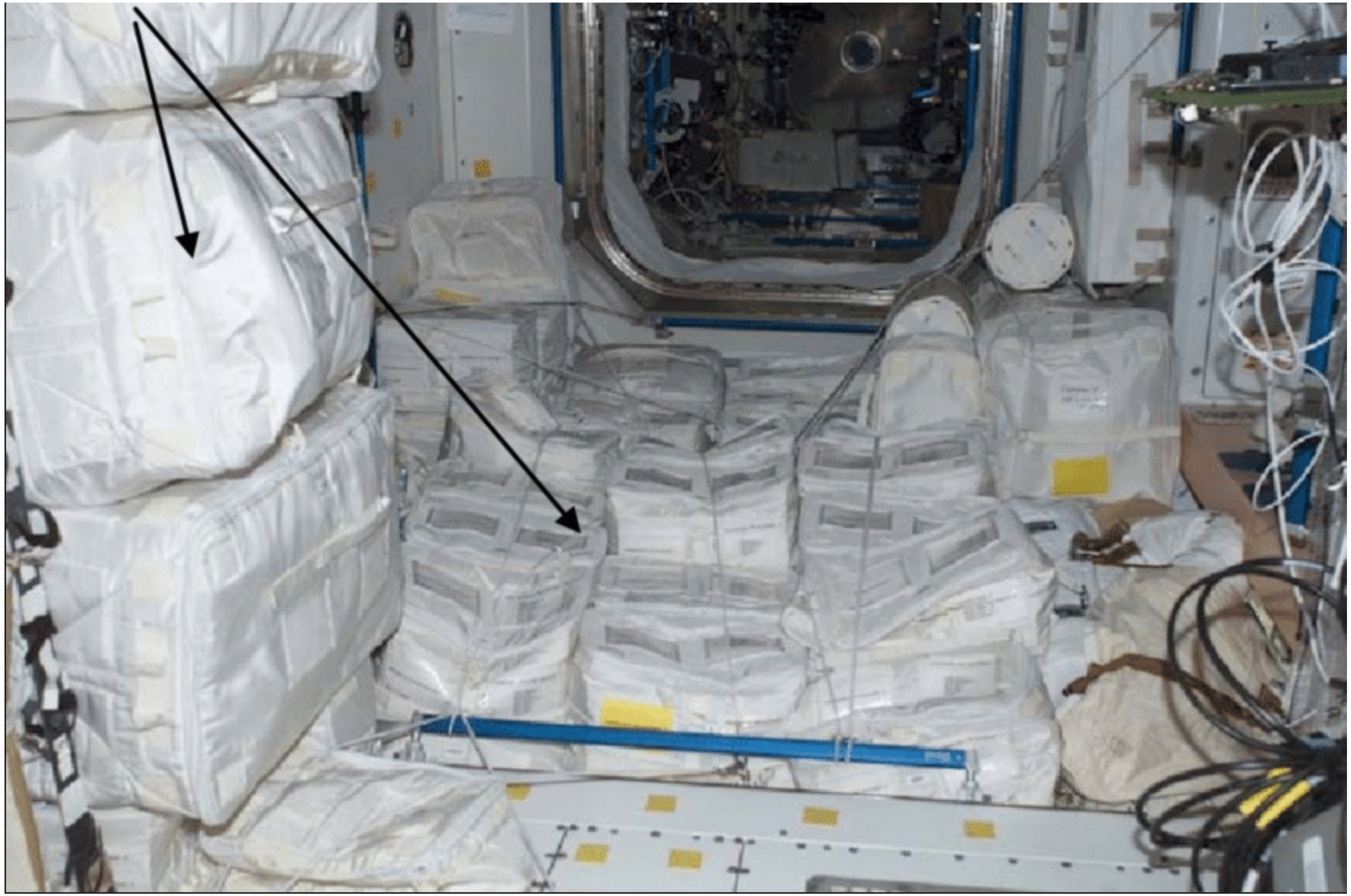


Janitor robot

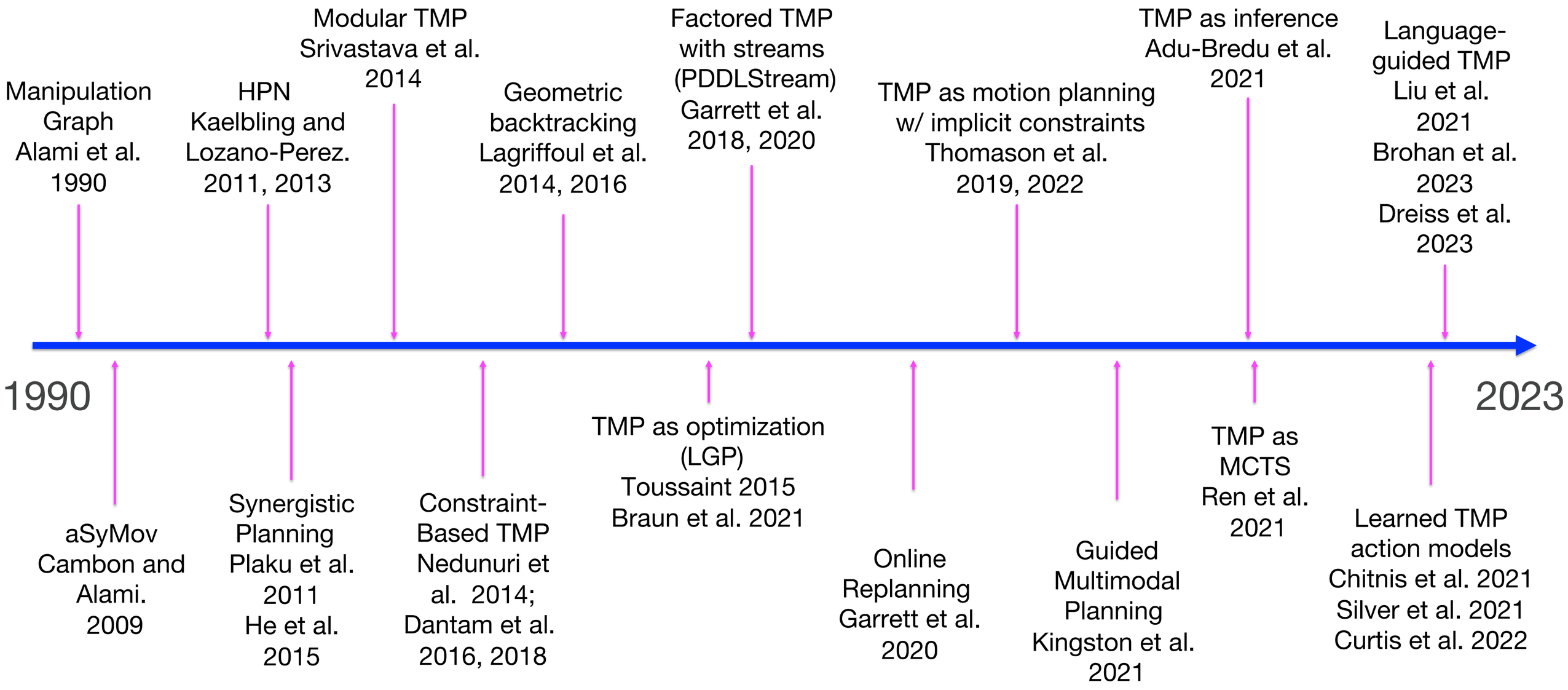
iRobot image:
http://i.imgur.com/Y0SITTQ.png

How do you Plan a Mission?

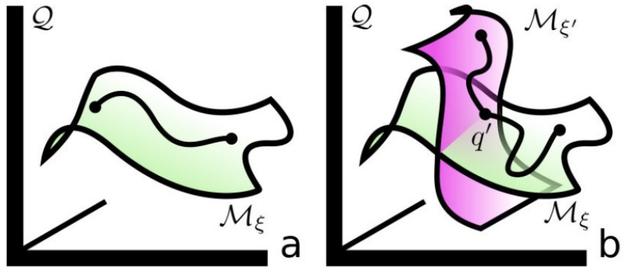




Task and Motion Planning (TAMP) - an Incomplete Timeline



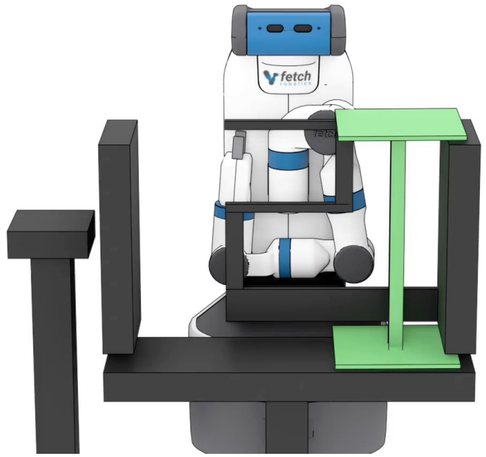
Known Environments – Manifold Constraints

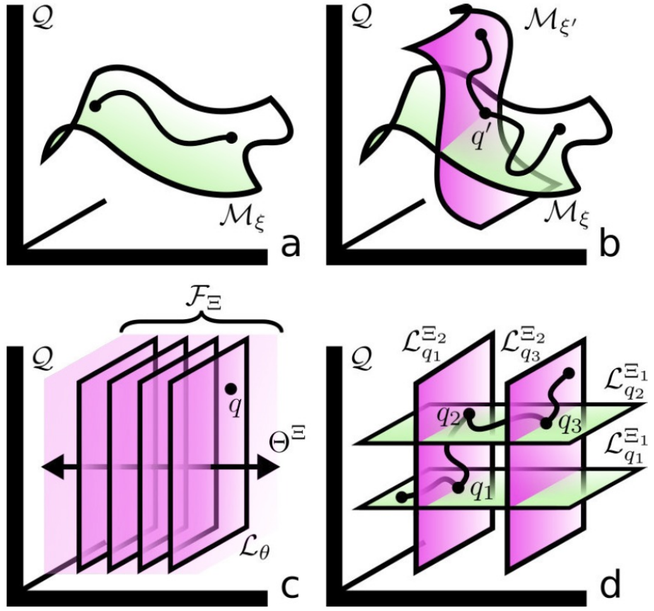


Guided Multimodal Motion Planning

Z. Kingston and L. E. Kavraki, “Scaling Multimodal Planning: Using Experience and Informing Discrete Search,” *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 128–146, Feb. 2023.

Known Environments – Manifold Constraints

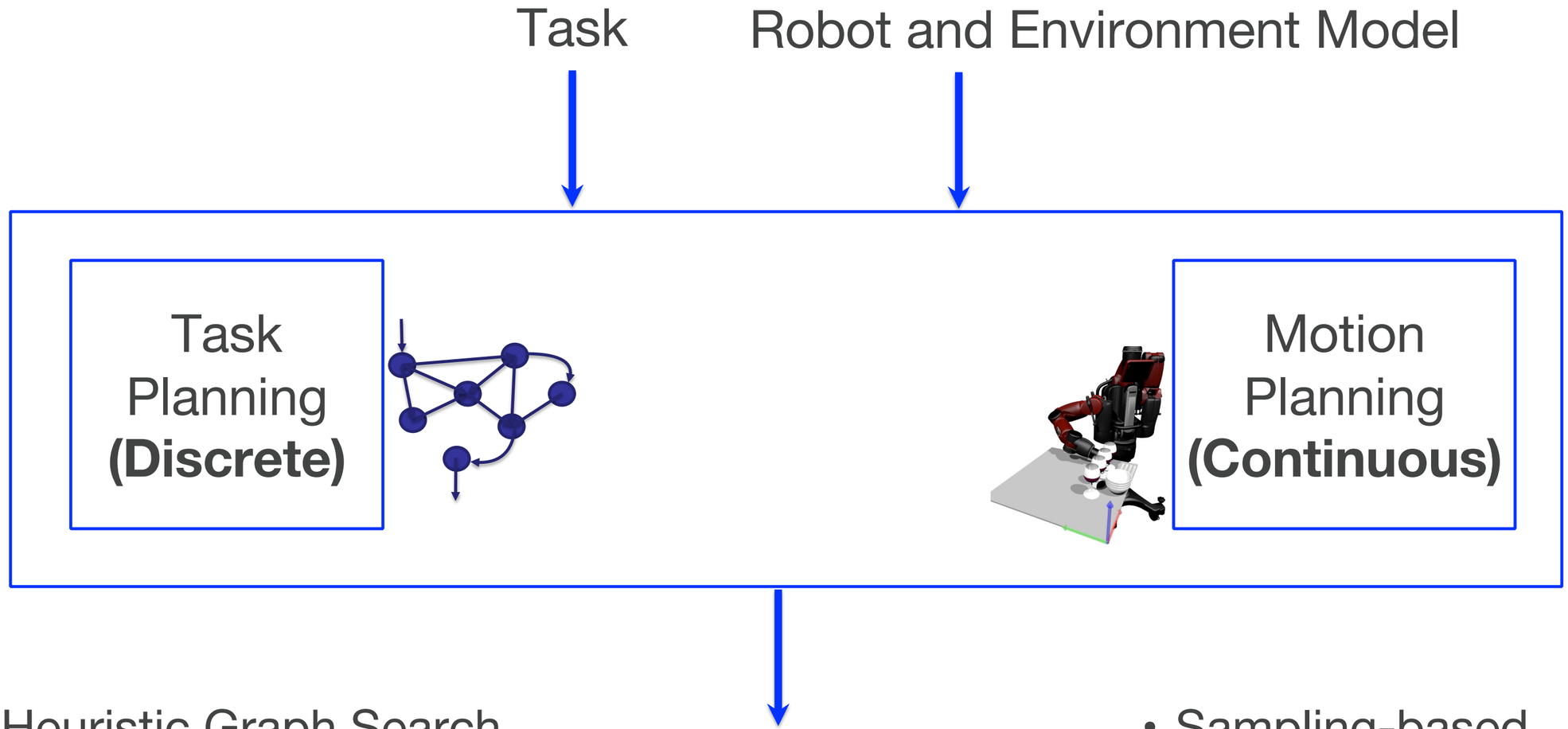




Guided Multimodal Motion Planning

Z. Kingston and L. E. Kavraki, “Scaling Multimodal Planning: Using Experience and Informing Discrete Search,” *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 128–146, Feb. 2023.

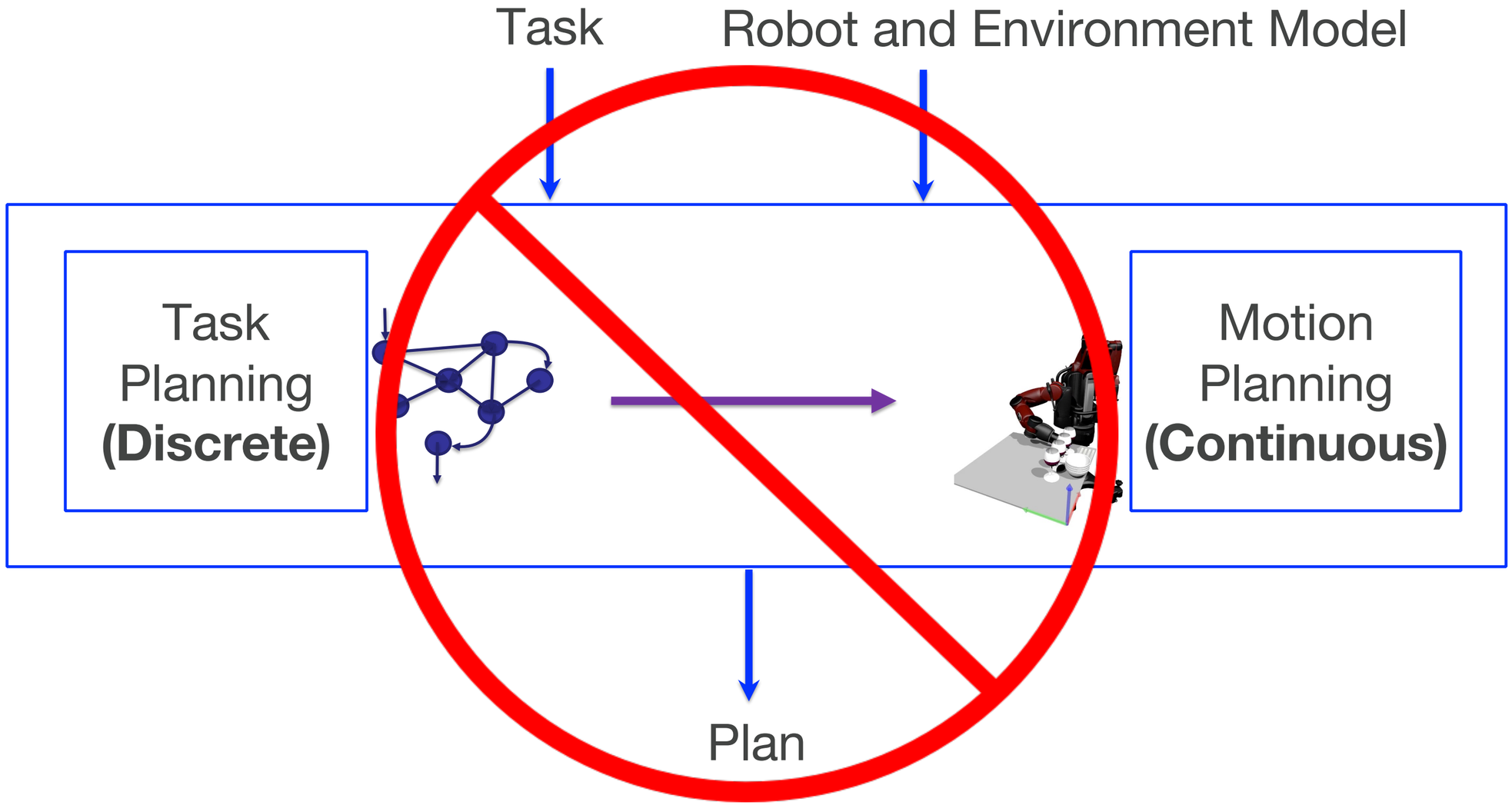
Known Environments – High Task Complexity



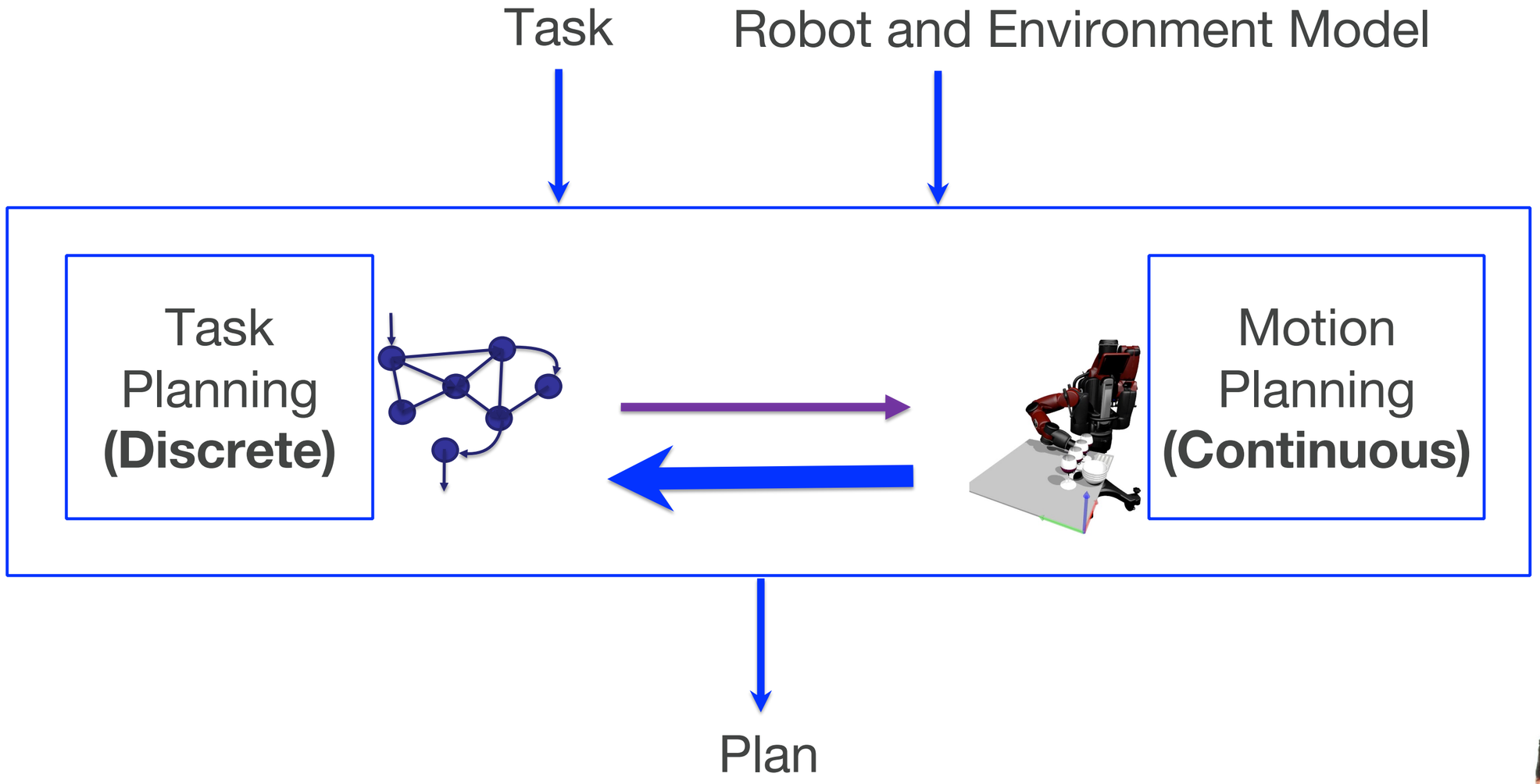
- Heuristic Graph Search
- Constraint Satisfaction
- Formal Methods
- Large Language Models

- Sampling-based
- Optimization-based
- Learning-based

Task and Motion Planning (TAMP)

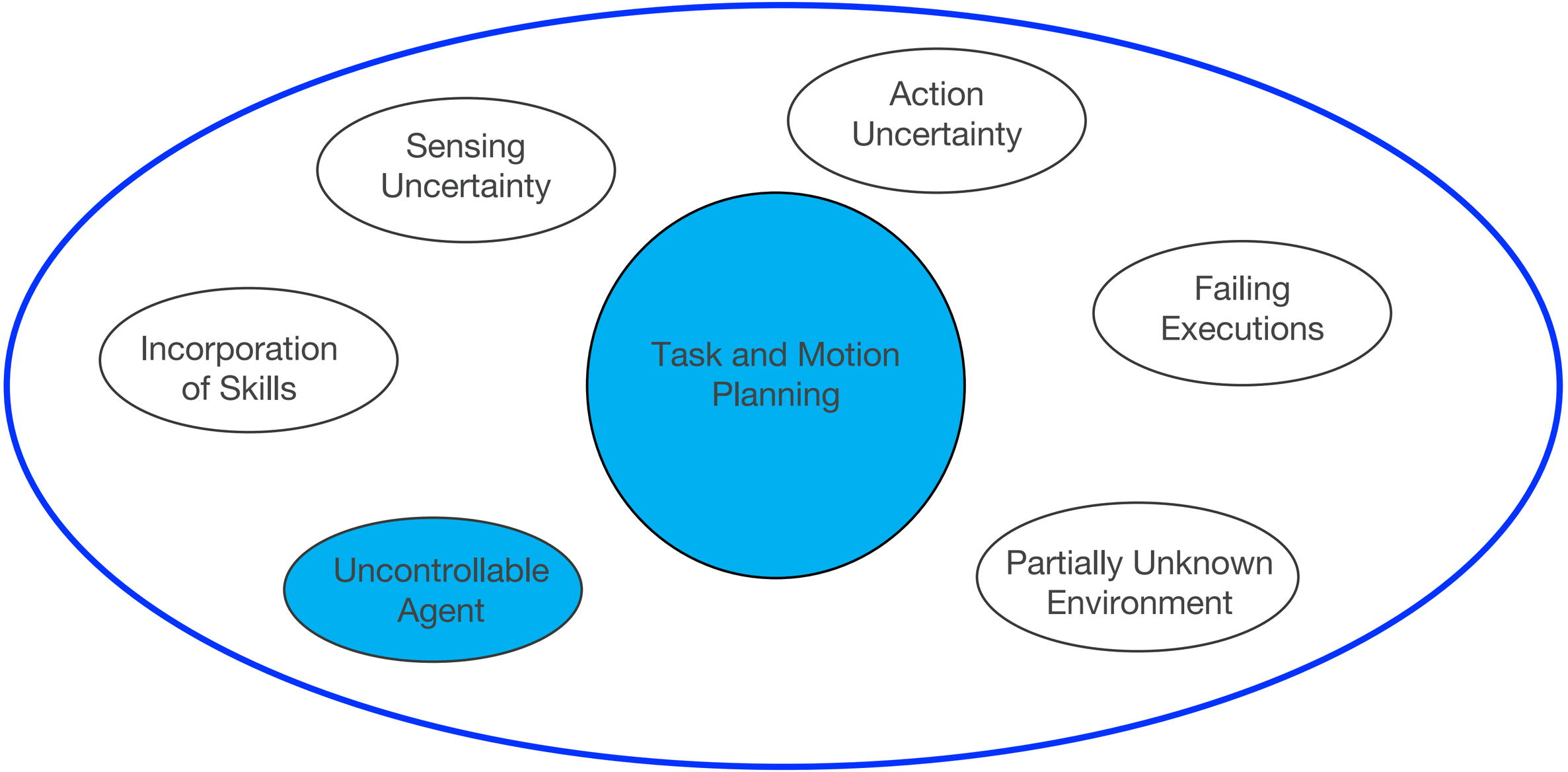


Incremental Constraint-Based Framework for TAMP

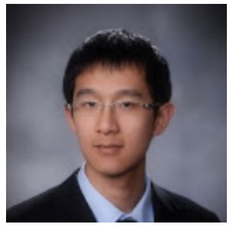


N. Dantam, Z. Kingston, S. Chaudhuri, L. Kavraki. *Incremental Task and Motion Planning: A Constraint-Based Approach*. IJRR 2018

From Motion Planning to Task and Motion Planning & Beyond



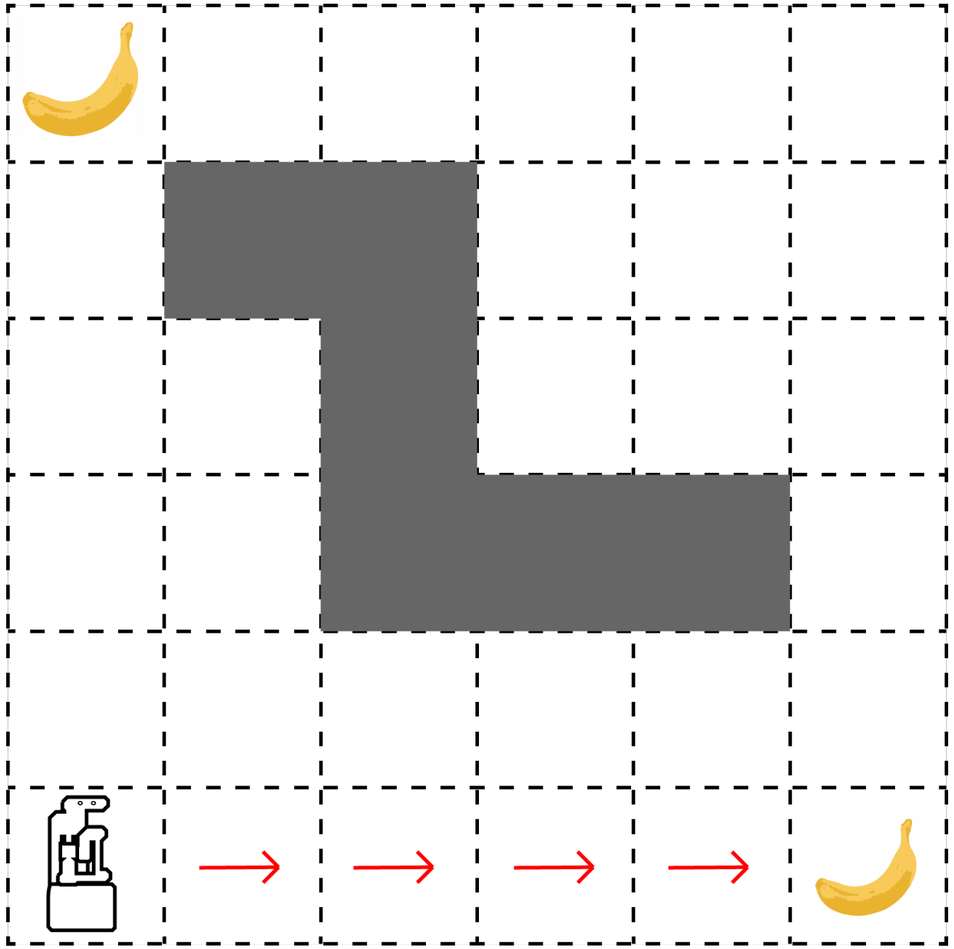
Reactive Robotic Tasks

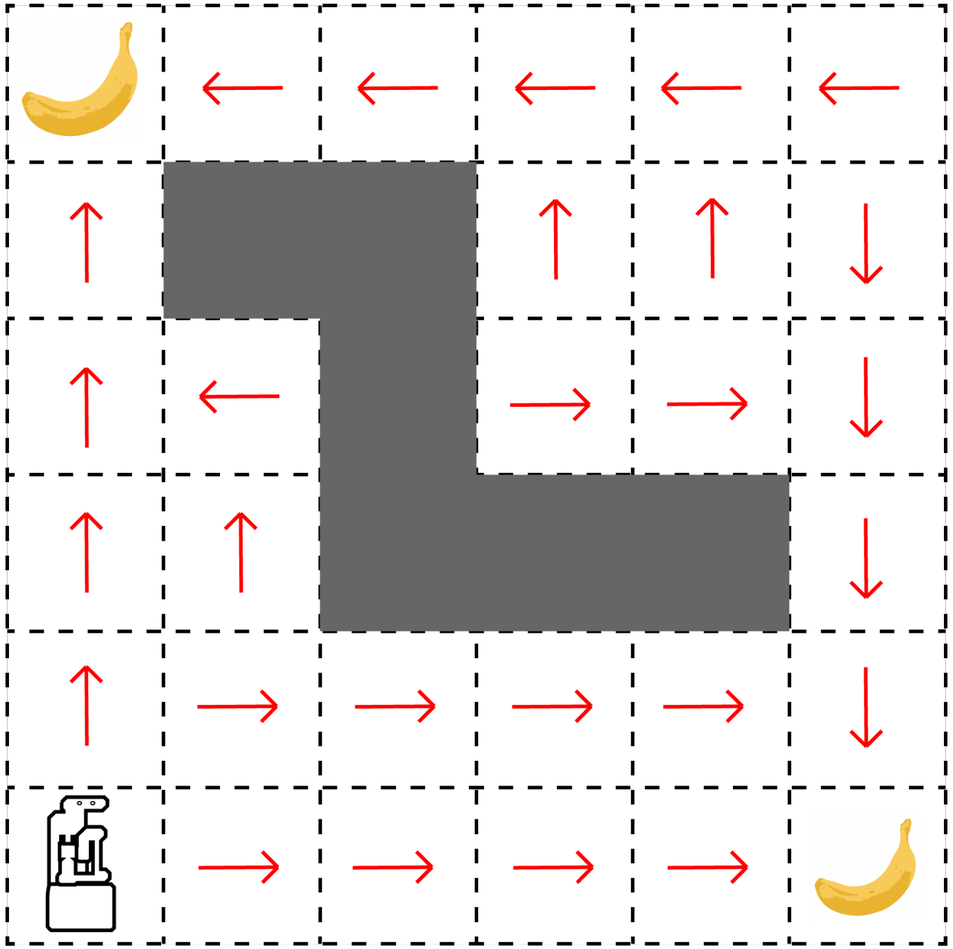


https://www.manufacturingtomorrow.com

Work in collaboration with M. Lahijanian and M. Vardi

Plans versus Policies





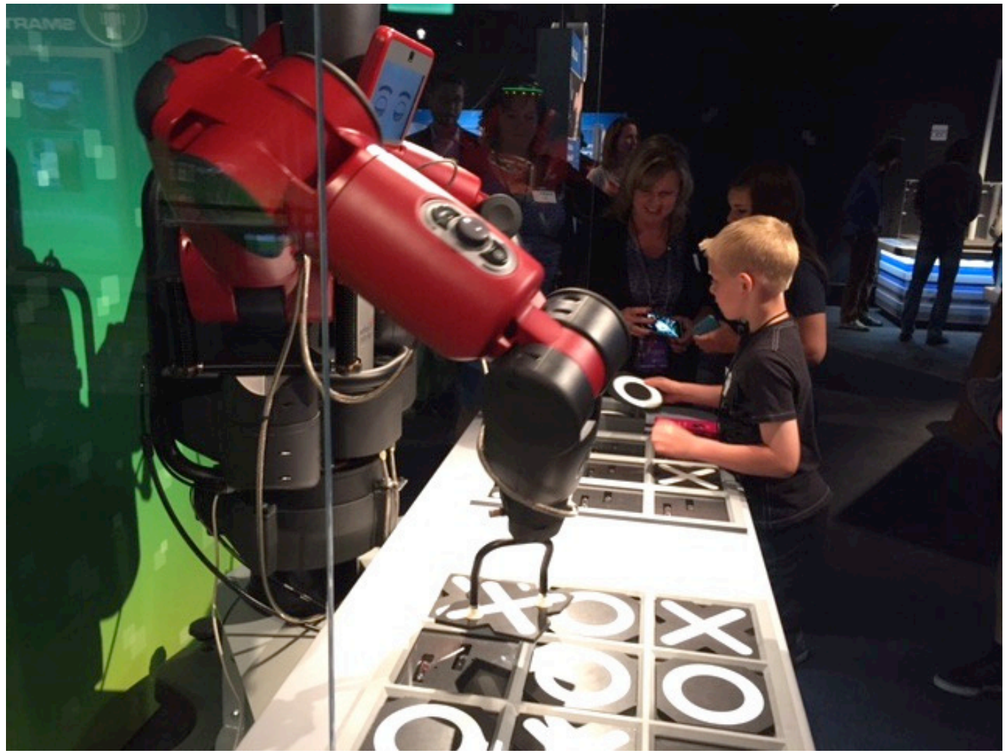
What Is Our Goal?

- Robot is given:
  - Some *task* to accomplish
  - Some *abstract model* of actions robot can take (and possible outcomes)
  - Some *abstract model* of the human
- Robot should output:
  - Actions to complete the task according to some criterion for optimality

Related Approaches

	Re-planning	Contingency Planning	Reactive Synthesis	Policy Learning
Task Completion Guarantee	With reversible actions or dead-end free domains	Yes* (if **complete** contingent plan constructed)	Yes	No
Runtime Latency	Poor (but improving)	Good	Good	Good
Scalability	Good	Poor	Okay	Okay* (horizon length)
Literature (an incomplete list)	Maly et al. 13 Guo et al. 13 Lahijanian et al. 16 Yang & Brock 10 Bowen & Alterovitz 14 Garrett et al. 20 Lim et al. 21 Castaman et al. 21 Pan et al. 22 Lager et al. 22	Peot & Smith 92 Pryor & Collins 96 Lahijanian et al. 14 Muise et al. 14 Santana et al. 16 Rizwan et al. 18 Shah et al. 20 Akbari et al. 20 Wang et al. 21	Kress-Gazit et al. 09, 11 Wongpiromsran et al. 11, 12 Wolff et al. 13 Vasile & Belta 14 Ehler & Raman 16 Moarref et al. 18 He et al. 18, 19 Wells et al. 21, 22	Levine et al. 15, 16, 18 Finn and Levine 16 Nair et al. 18 Zhu et al. 18 Gupta et al. 19 Singh et al. 19 Ichter et al. 20 Sharma et al. 20 Shridhar et al. 22 Wang et al. 22 Janner et al. 22 Brohan et al. 22, 23

Reactive Synthesis for Robots

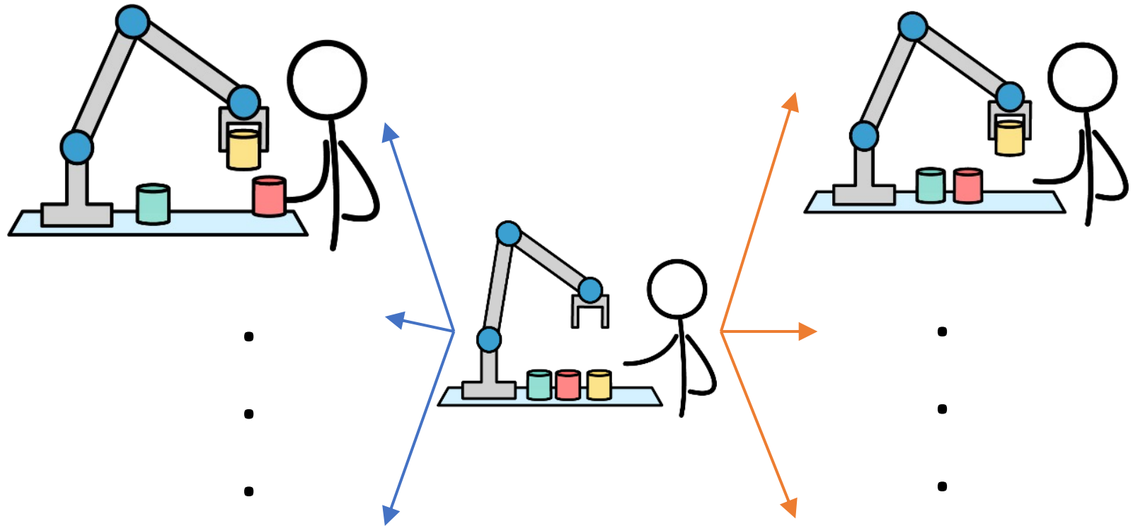


https://www.redtri.com

Game		Reactive Synthesis
Players	———	Robot and Human
Game Rules	———	Planning Domain (Abstraction)
Winning Condition	———	Task (Specification)
Strategy that Guarantees Robot Wins	———	Strategy that Guarantees Task Completion

[Kress-Gazit et al. 07] [Wongpiromsarn et al. 10] [Wolff et al. 13]
[Vasile & Belta 14] [Ehler & Raman 16] [He et al. 17]

Two Elements of Reactive Synthesis

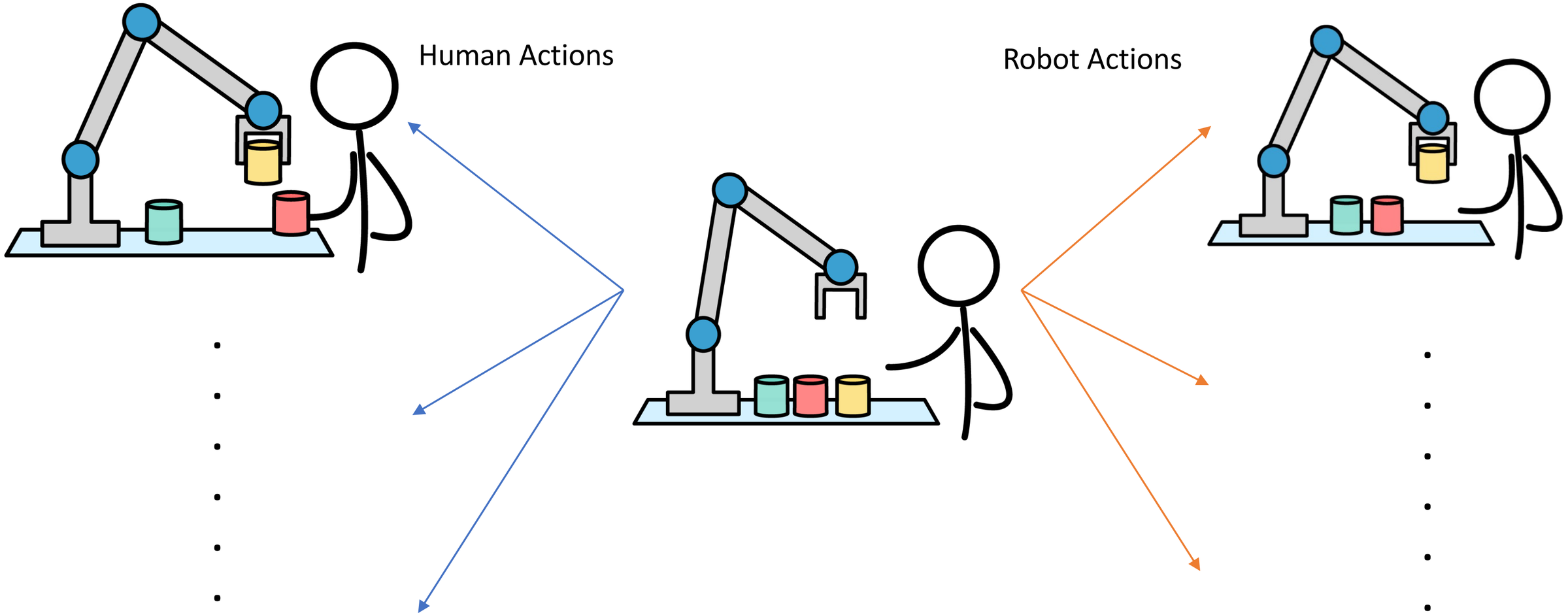


Domain Abstraction

LTL_f Task Specification

Find a policy that guarantees task completion

Domain Abstraction of Robot Manipulation



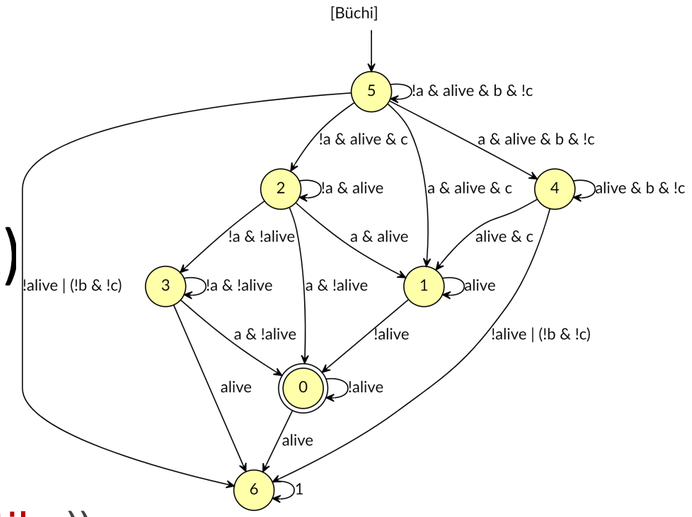
We restrict the actions of the human

K. He, M. Lahijanian, L.E. Kavraki, and M.Y. Vardi, Automated Abstraction of Manipulation Domains for Cost-Based Reactive Synthesis, IEEE RAL, 2019

Reactive Synthesis Specification: LTL_f

- Linear Temporal Logic on Finite Traces (LTL_f)
  - Subsumes Boolean logic: Boolean variables + operators: And, Or, Not, Implies
  - Adds temporal operators: Next, Until, Eventually, Always

- Suitable for robotics tasks (e.g., construction)
- Can be translated to a deterministic finite automaton (DFA)



(Eventually a And (b Until c))

Automaton generated by SPOT
[Duret-Lutz et al., 2016]

De Giacomo, G., Vardi, M.Y.: Linear temporal logic and linear dynamic logic on finite traces. In IJCAI. vol. 13, pp. 854–860 (2013)
De Giacomo, G., Vardi, M.Y.: Synthesis for LTL and LDL on finite traces. In IJCAI. vol. 15, pp. 1558–1564 (2015)

Efficient Symbolic Reactive Synthesis for Finite-Horizon Tasks

Keliang He, Andrew Wells, Lydia E. Kavraki, Moshe Y. Vardi
Rice University

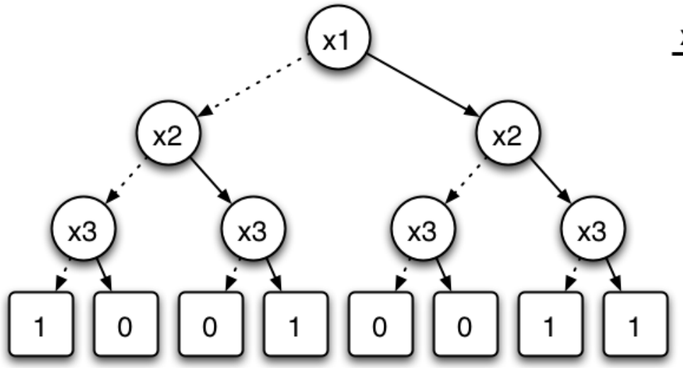
Efficient Symbolic Reactive Synthesis for Finite-Horizon Tasks

Keliang He, Andrew Wells, Lydia E. Kavraki, Moshe Y. Vardi
Rice University

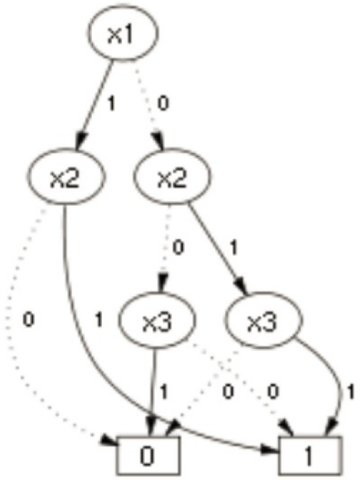
Symbolic Synthesis via Binary Decision Diagrams

- Two main parts to our problem: Transition System and DFA
- Both given with respect to some Boolean variables
- Our computations are concerned with Boolean functions and sets
- Binary Decision Diagrams (BDDs) compactly represent Boolean functions and sets

BDDs:



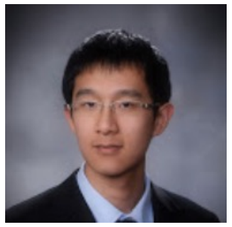
x1	x2	x3	f
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

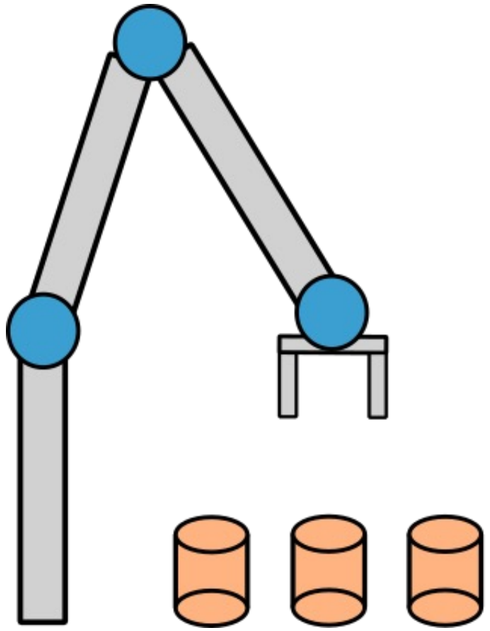


Example from Wikipedia

R. E. Bryant, “Graph-based algorithms for Boolean function manipulation,” IEEE Transactions on Computers, vol. 100, no. 8, pp. 677–691, 1986.

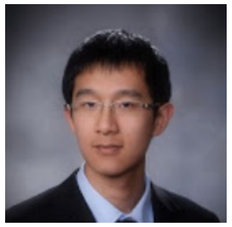
Symbolic Synthesis

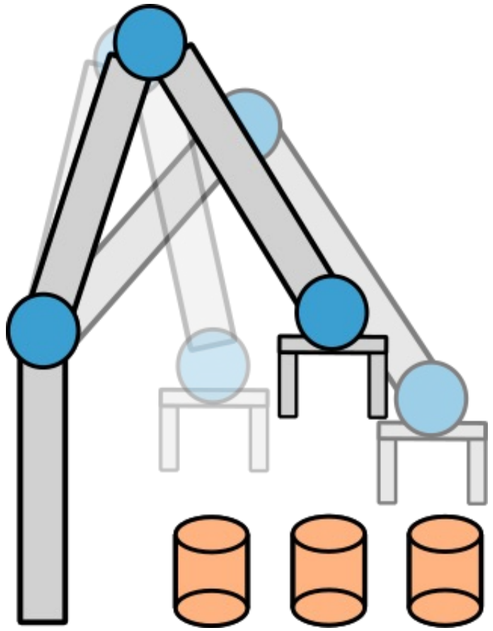




- The robot needs to grasp a cylinder
- Rather than dealing with all states explicitly, we would like to group “equivalent” states together

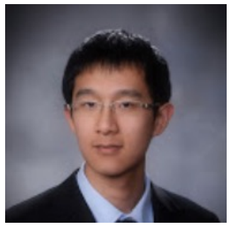
Symbolic Synthesis



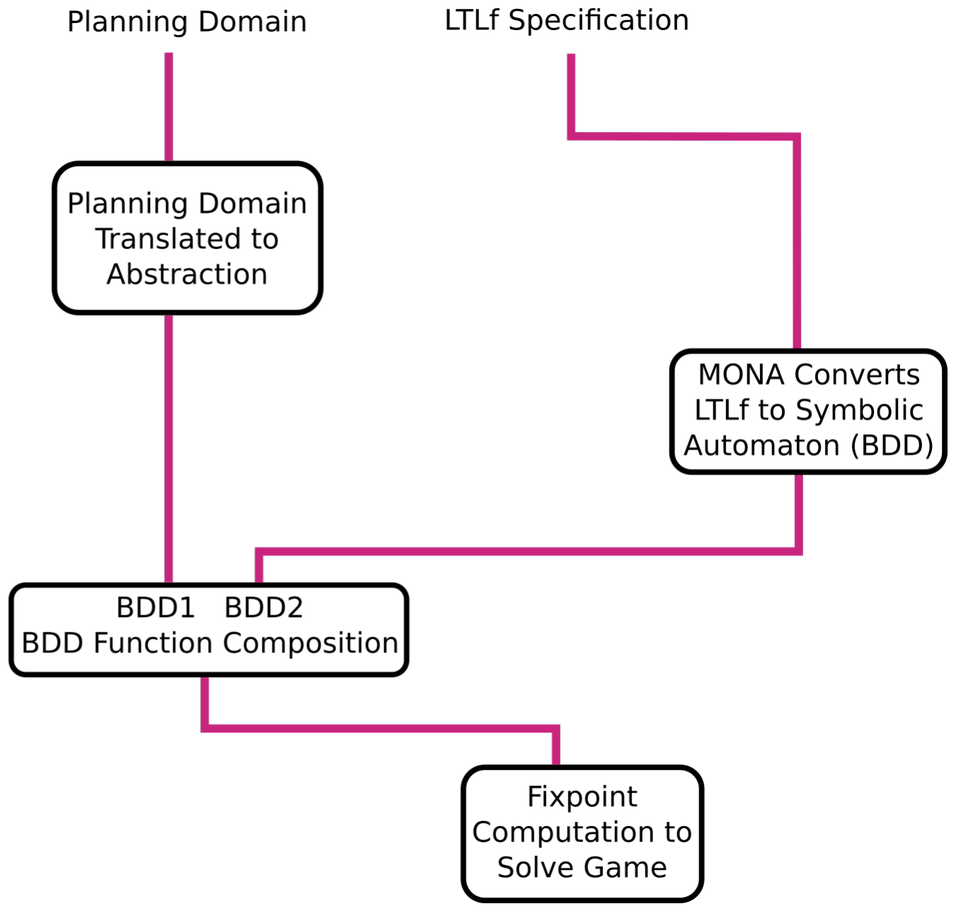


- The only state variable we care about is that some cylinder is in the gripper
- The synthesis method will see that all three of these options will satisfy this goal
- For our task, these states are “equivalent”

Scalability through Symbolic Representations



K. He, Ph.D. Thesis



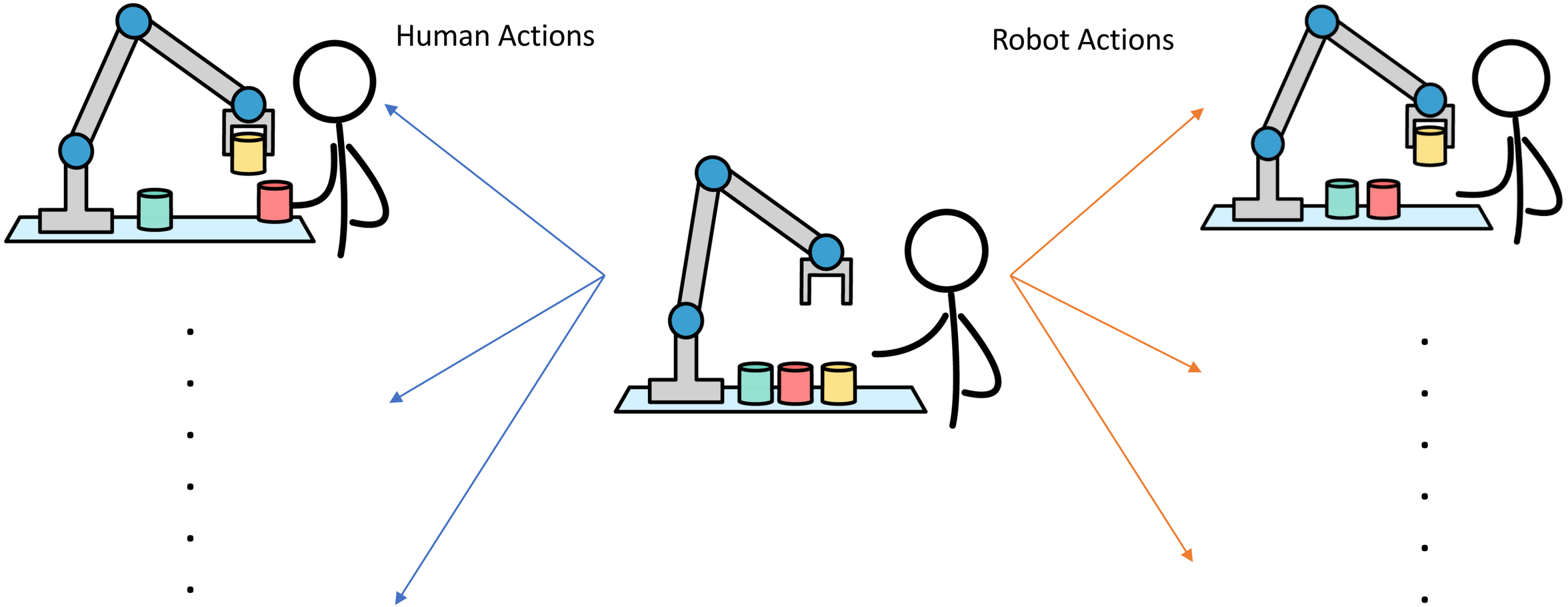
Henriksen et al. “Mona: Monadic second-order logic in practice” TACAS 1995
Bryant. “Graph-Based Algorithms for Boolean Function Manipulation” IEEE Transactions on Computers 1986
Zhu et al. “Symbolic LTLf Synthesis” AAAI 2017

From Reactive Synthesis to Probabilistic Synthesis

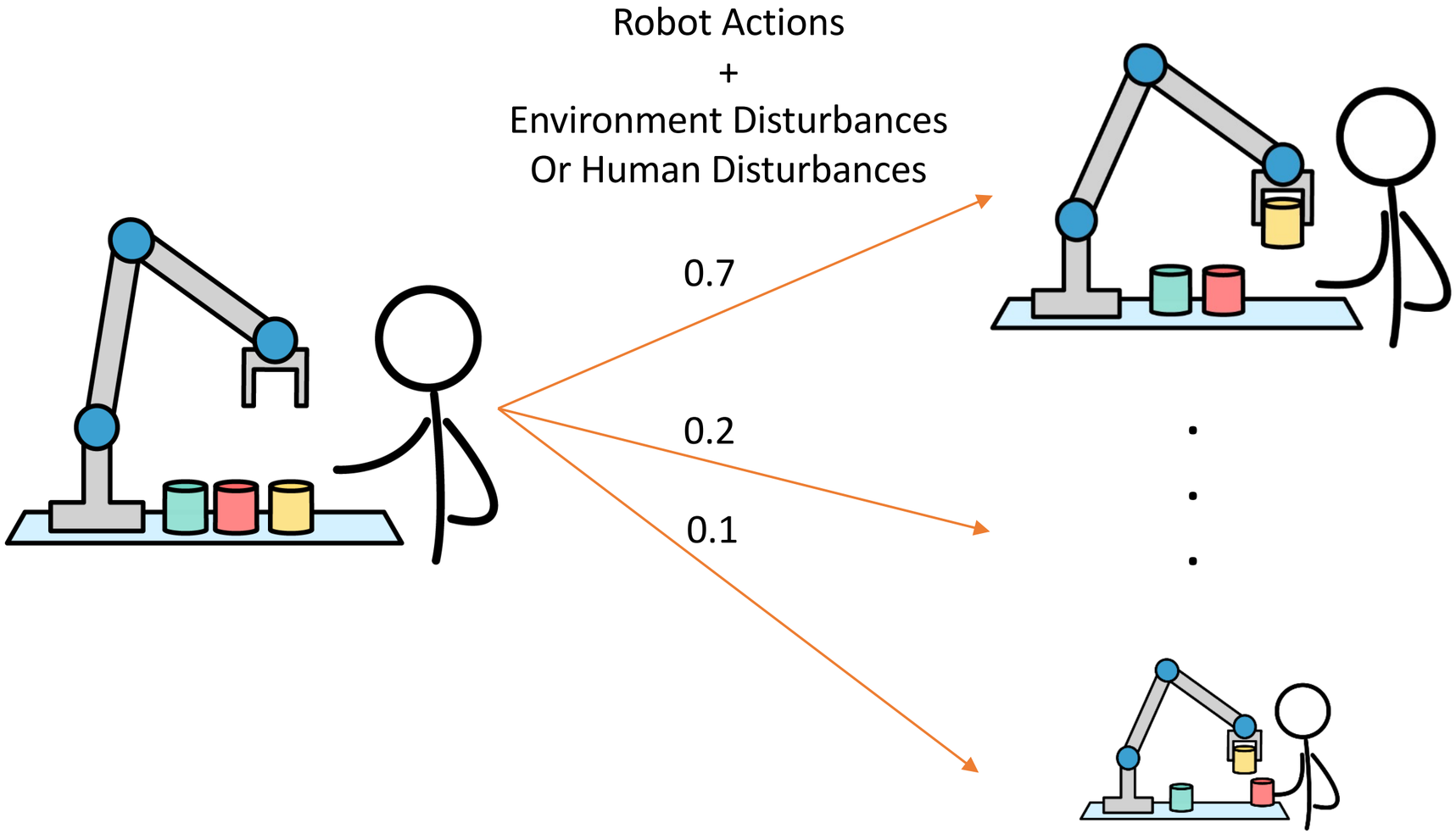
- Our Reactive Synthesis:
  - Improved state-of-the-art scalability
  - World model is still inaccurate
- Desired Approach:
  - Model the world more closely
  - Keep (and eventually improve) scalability

A. Wells, M. Lahijanian, L.E. Kavraki, M.Y. Vardi. Finite Horizon Synthesis for Probabilistic Manipulation Domains, ICRA 2021

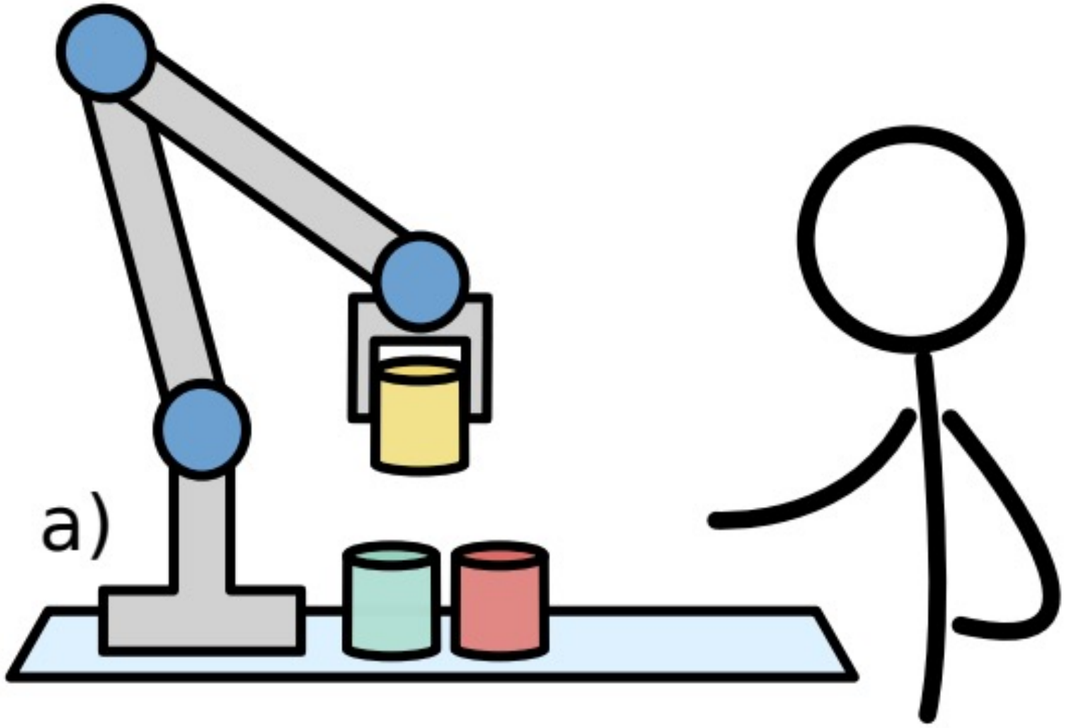
Domain Abstraction for Reactive Synthesis



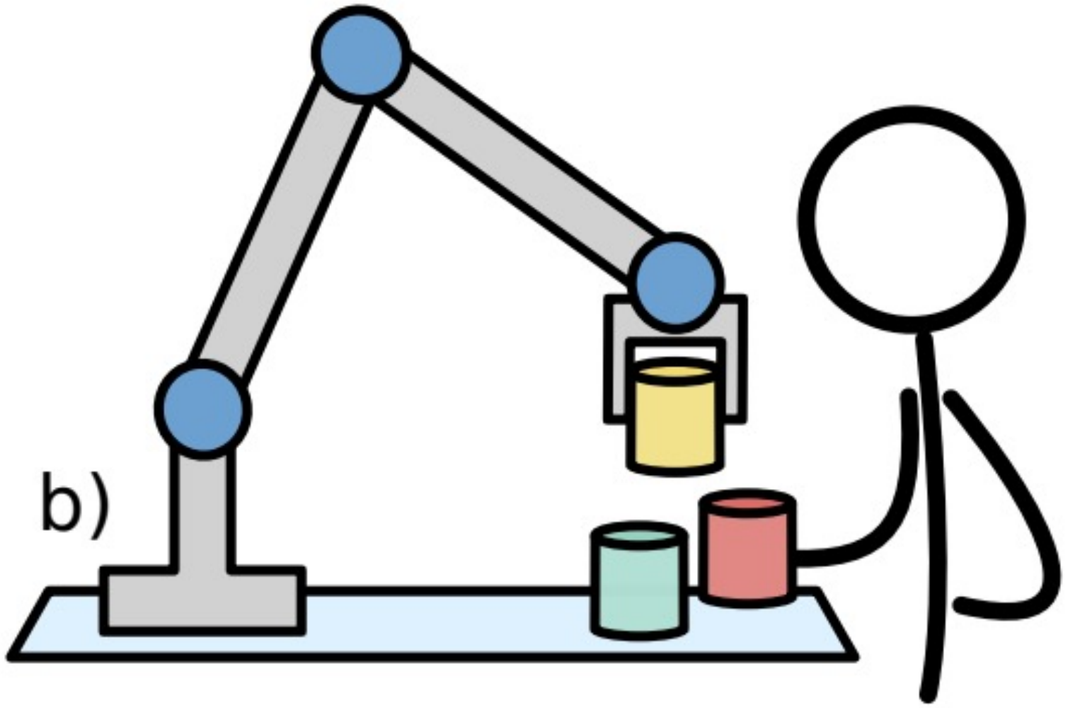
Domain Abstraction for Probabilistic Synthesis



Modeling the Human

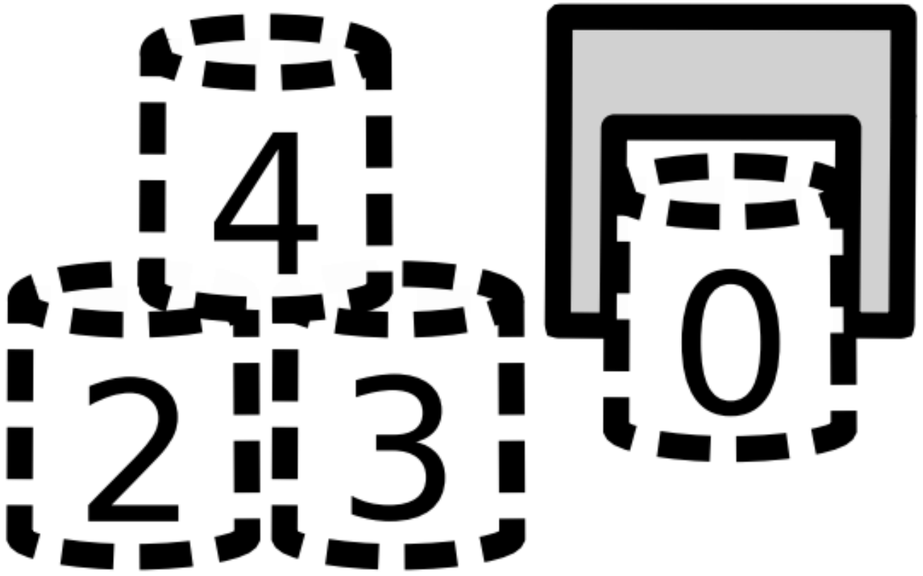


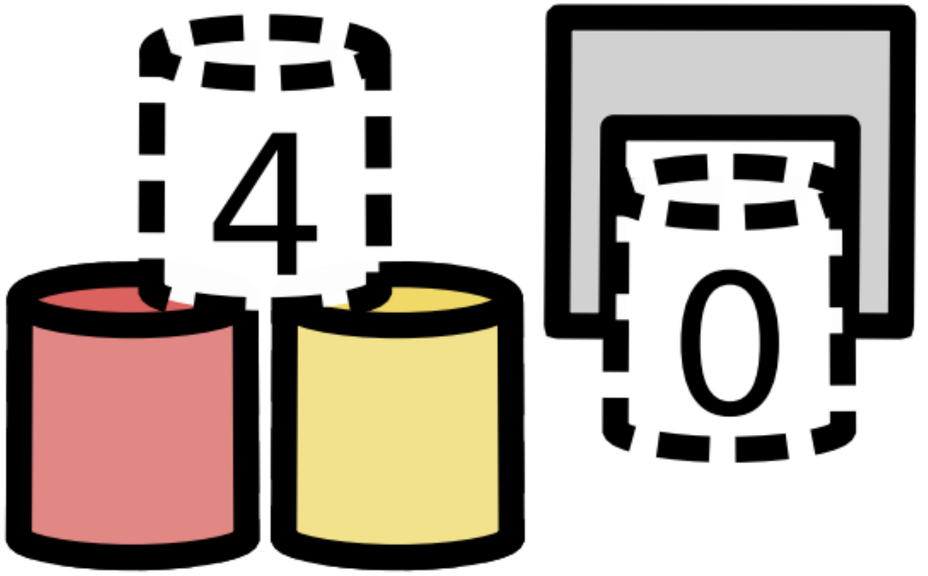
Adversarial human



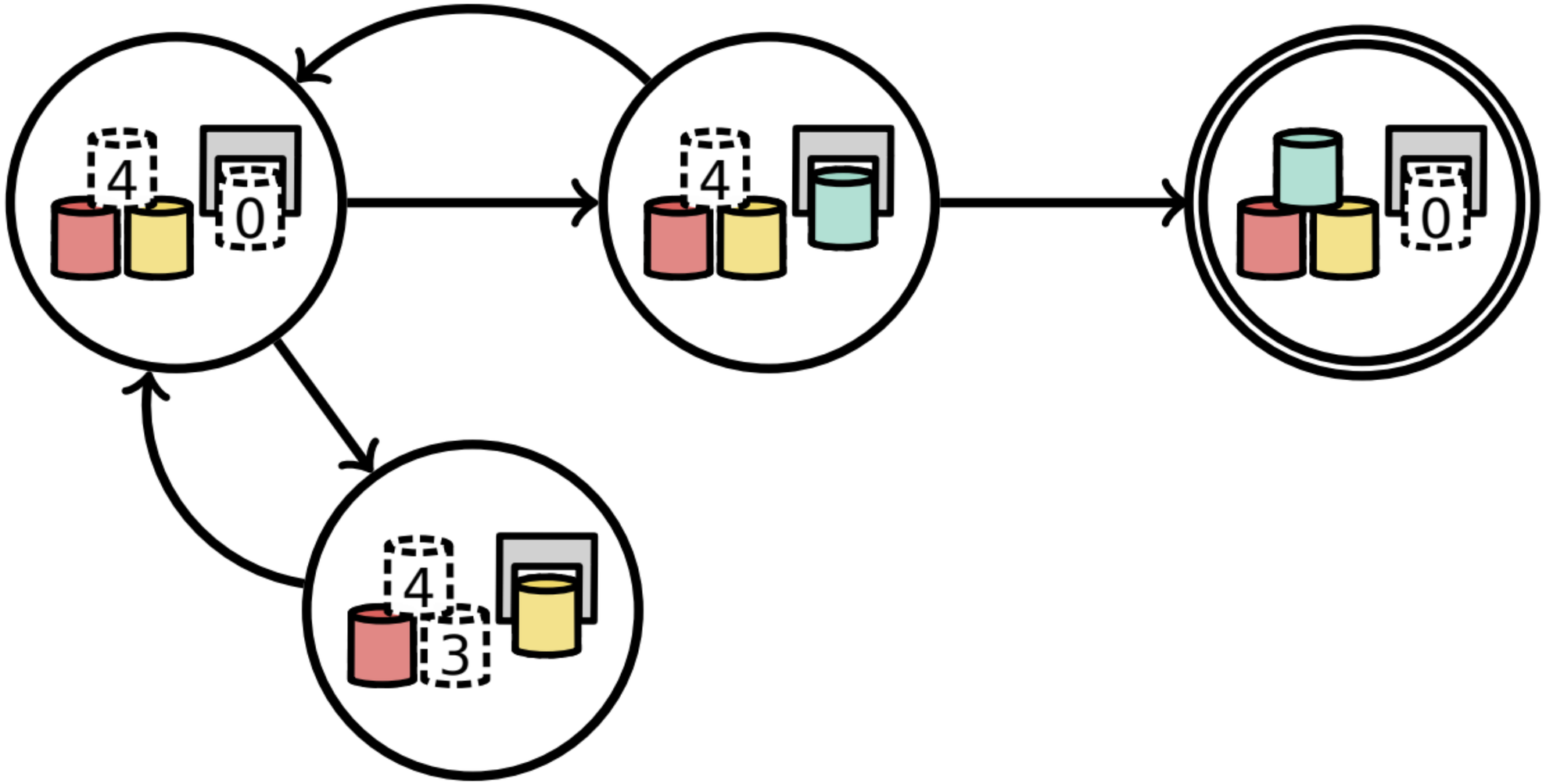
Helpful human

Formalizing Human-Robot Manipulation

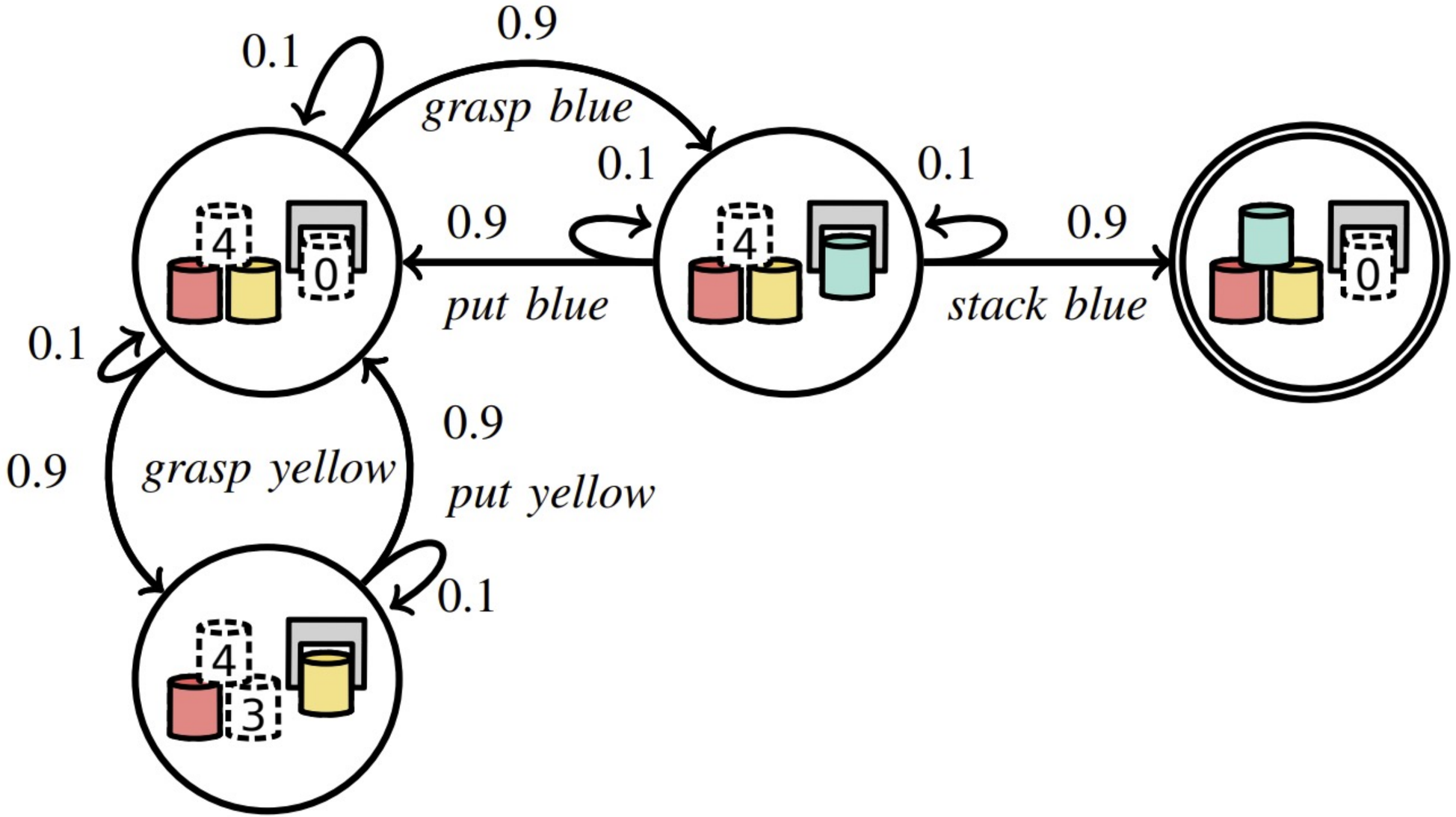




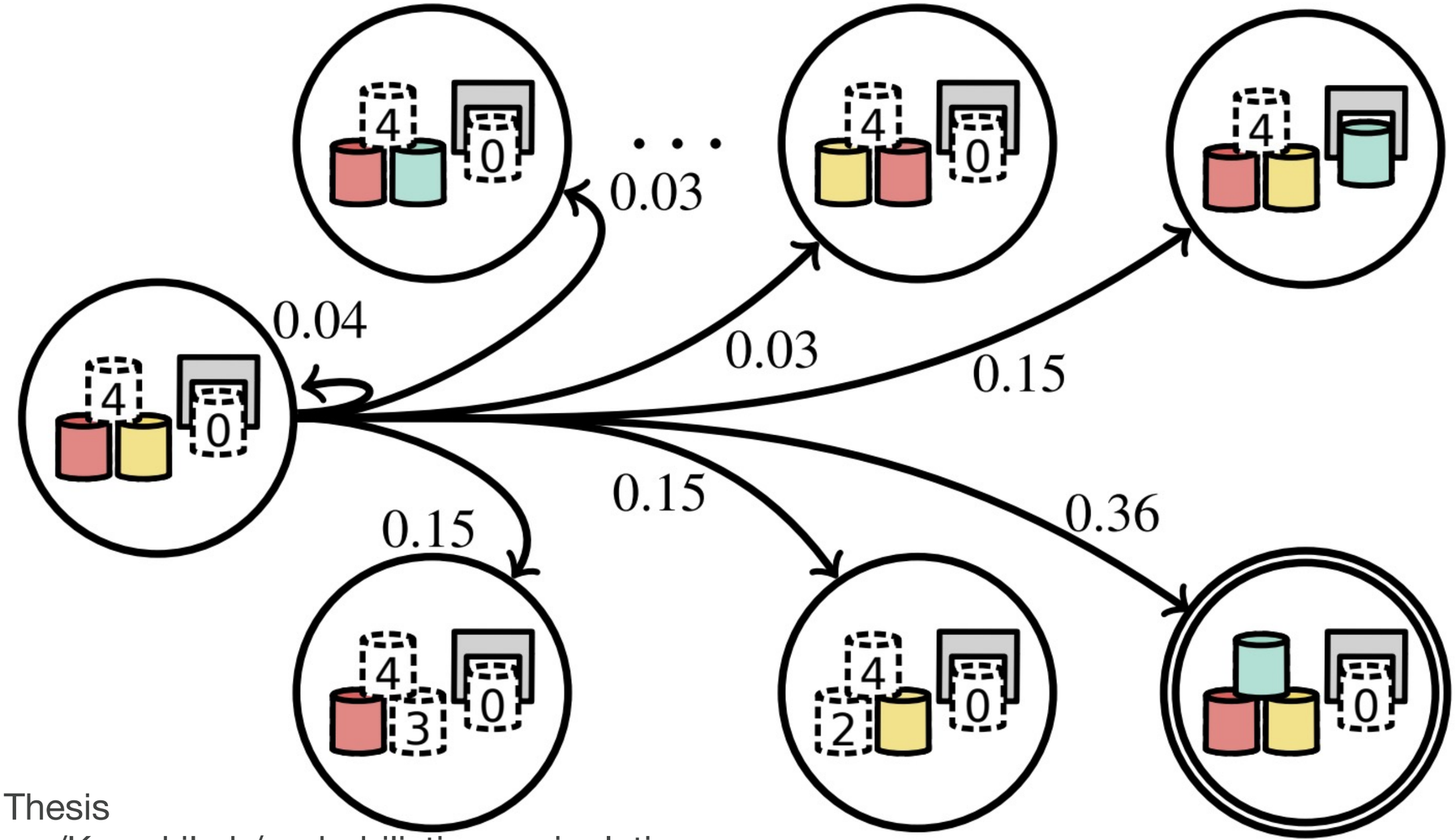
Formalizing Human-Robot Manipulation



Robot Distribution



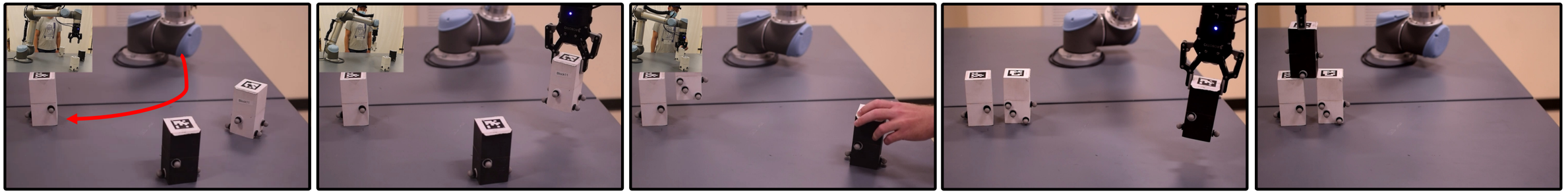
Formalizing Human-Robot Manipulation: Constructing a MDP_{manip}



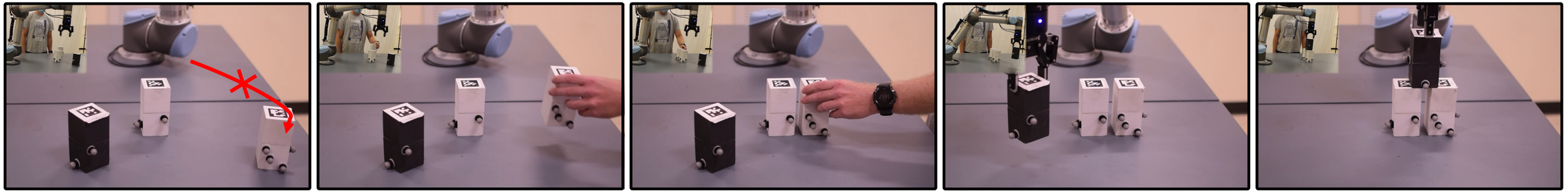
A. Wells, Ph.D. Thesis
https://github.com/KavrakiLab/probabilistic_manipulation

Example

We do not expect human collaboration (we may have an adversarial human):
the robot builds the arch away from the human



We expect human collaboration:
the robot builds the arch close to the human



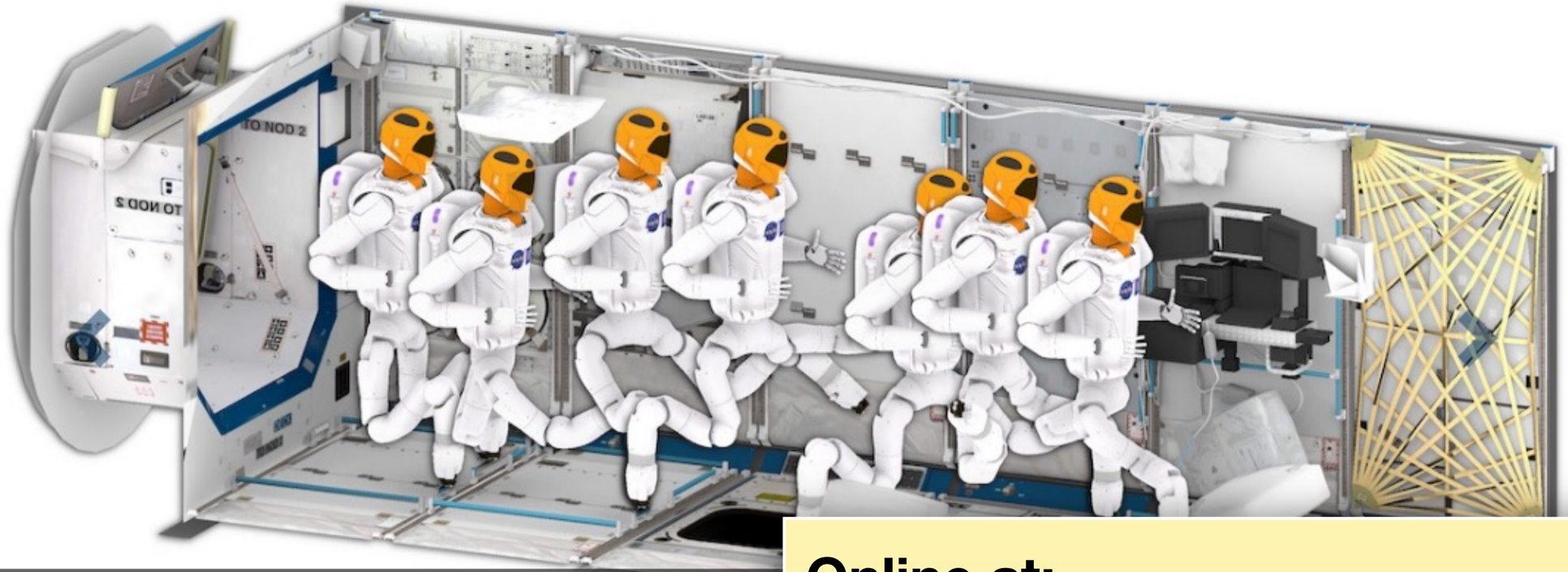
Task and Motion Planning (TAMP)

- Pay attention to Abstractions / Representations
- Exploit Modularity / Hierarchies
- Find the right place for formal methods, classical AI planning, learning of all sorts
- Work on the meaningful interaction of perception, state estimation, planning and control

Broader Impact

OMPL Download Documentation Gallery OMPL Integrations Code Issues Community About Blog Search

The Open Motion Planning Library



OMPL used in ROS/MoveIt

OMPL is the default planning library in MoveIt and has been used for many robots. Here, OMP

OMPL, the Open Motion Planning Library, consists of many state-of-the-art sampling-based moti
contain any code related to, e.g., collision checking or visualization. This is a deliberate design ch
collision checker or visualization front end. The library is designed so it can be easily integrated i
needed components.

OMPL.app, the front-end for OMPL, contains a lightweight wrapper for the FCL and PQP collisio
PySide. The graphical front-end can be used for planning motions for rigid bodies and a few vehicle types (first-order and second-order cars, a
blimp, and a quadrotor). It relies on the Assimp library to import a large variety of mesh formats that can be used to represent the robot and its
environment.

Online at:
https://ompl.kavrakilab.org

Public repository at:
https://github.com/ompl

Some Sampling-Based Motion Planners in OMPL

Multi-query Planners

- **PRM** (Kavraki et al 1996)
- **LazyPRM** (Bohlin & Kavraki 2000)
- **QMP** (Orthey et al. 2018)

Single-Query Planners

- **RRT** (Kuffner & Lavalle 2000)
- **RRTConnect** (Kuffner & Lavalle 2000)
- **EST** (Hsu et al 1999)
- **SBL** (Sancez & Latombe 2001)
- **KPIECE** (Sucan & Kavraki 2008)
- **STRIDE (**Gibson et al. 2013)
- **PDST** (Ladd & Kavraki 2005)
- **QRT** (Orthey & Toussaint 2019)

Kinodynamic Planners

- **RRT** (Kuffner & Lavalle 2000)
- **EST** (Hsu et al 1999)
- **KPIECE** (Sucan & Kavraki 2008)
- **Syclop** (Plaku et al. 2010)
- **PDST** (Ladd & Kavraki 2005)
- **Sparse Stable RRT** (Li et al. 2016)

Constrained Planners

- **CBiRRT2** (Berenson et al. 2011)
- **AtlasRRT** (Jaillet & Porta 2013)
- **Tangent Bundle RRT** (Kim et al. 2016)
- **General Manifold Constraints** (Kingston et al. 2019)

Optimizing Planners

- **CFOREST** (Otte & Correl 2013)
- **Anytime Path Shortening** (Luna et al. 2013)
- **RRT*** (Karamazon & Frazzoli 2011)
- **PRM*** (Karaman & Frazzoli 2011)
- **SPARS** (Dobson et al. 2012)
- **SPARS2** (Dobson & Bekris 2013)
- **Transition-Based RRT** (Jaillet et al. 2000)
- **LBTRRT** (Salzman & Halperin 2013)
- **Sparse Stable RRT** (Li et al. 2016)
- **Vector Field RRT** (Ko et al. 2014)
- **ST-RRT*** (Grothe et al. 2022)
- **RRT#** (Arslan & Tsiotras 2015)
- **RRTx** (Otte & Frazzoli 2015)
- **InformedRRT*** (Gammel et al. 2014)
- **FMT*** (Janson et al. 2015)
- **BFMT*** (Starek et al. 2015)
- **BIT*** (Gammel et al. 2015)
- **AIT*** (Strub and Gammel 2022)
- **EIT*** (Strub and Gammel 2022)

OMPL@Rice

- Developers at Rice:
  M. Moll, I. Sucan, Z. Kingston,
  C. Voss, R. Luna, M. Maly, N. Dantam, B. Wiley, C. Chamzas, T. Pan, C. Quintero-Pena, Wil Thomason

- Work on earlier packages by E. Plaku, K. Bekris, A. Ladd

Several contributors outside Rice!

OMPL Contributors

http://ompl.kavrakilab.org/developers.html

- **Jennifer Barry**, Leslie Pack Kaelbling and Tomás Lozano-Pérez's **Learning in Intelligent Systems Group**, MIT (now at Rethink Robotics)
- **Prudhvi Boyapalli**, Rice University
- Leonard Bruns, Robert Bosch GmbH (now at KTH Royal Institute of Technology)
- Stephen Butler, Rice University
- Beck Chen, Rice University
- **Sachin Chitta**, SRI International (now at Boston Dynamics)
- **Ashley Clark**, Steve Rock's **Aerospace Robotics Lab**, Stanford University
- **Dave Coleman**, Nikolaus Correll's **group**, University of Colorado Boulder (now at **PickNik Robotics**)
- **Neil Dantam**, Rice University (now at Colorado School of Mines)
- **Andrew Dobson**, Kostas Bekris' **Physics-aware Research for Autonomous Computational SYStems group**, Rutgers University
- Elizabeth Fudge, Rice University
- **Jonathan Gammell**, Tim Barfoot's **Autonomous Space Robotics Lab**, University of Toronto (now professor at Oxford)
- Bryant Gipson, Rice University (now at Google)
- **Javier V Gomez**, Universidad Carlos III de Madrid (now at Magic Leap)
- Florian Hauer, Georgia Tech
- Gil Jones, Google
- **Sertac Karaman**, Emilio Frazzoli's **Aerospace Robotics and Embedded Systems Laboratory**, MIT
- Henning Kayser, **PickNik Robotics**
- **Zachary Kingston**, Rice University
- **Ryan Luna**, Rice University (now at Waymo)
- Matt Maly, Rice University (now at Google)
- **James Marble**, Kostas Bekris' **Physics-aware Research for Autonomous Computational SYStems group**, University of Nevada, Reno
- **Andreas Orthey**, Max-Planck Institute for Intelligent Systems, Marc Toussaint's **Intelligent & Learning Systems Lab**, Technical University of Berlin
- Luigi Palmieri, Robert Bosch GmbH
- Scott Paulin, University of Canterbury, New Zealand
- **Alejandro Perez**, Seth Teller's **Robotics, Vision, and Sensor Networks Group**, MIT
- **Oren Salzman**, Dan Halperin's **Computational Geometry Lab**, Tel Aviv University
- Edward Schmerling, Marco Pavone's **Autonomous Systems Lab**, Stanford University
- Jonathan Sobieski, Rice University
- Marlin Strub, **Estimation, Search, and Planning group** at Oxford
- Sonny Tarbouriech, University of Sherbrooke
- **Luis Torres**, Ron Alterovitz' **Computational Robotics Group**, University of North Carolina at Chapel Hill (now at Google)
- **Caleb Voss**, Rice University (now at Georgia Tech)
- Bryce Willey, Rice University (now at **Realtime Robotics**)

The OMPL github repository lists 69 contributors

OMPL Metrics 2023

- Estimated 3,500 users
- OMPL web site since January 2011:
  - 366,577 unique visitors
  - 8,427,036 page views, 4 pages/visit, 6 minutes per visit
- Available in many package managers for all common operating systems:
  - apt (Ubuntu/Debian)
  - MacPorts/HomeBrew (macOS)
  - vcpkg (Microsoft Windows)
- Widespread support:

MoveIt

OpenRAVE

Habitat

CoppeliaSim
from the creators of V-REP

MORSE
the Modular OpenRobots Simulation Engine

Robotics Library

VEROSIM SOLUTIONS

OMPL

Online at:
https://ompl.kavrakilab.org

Public Repository:
https://github.com/ompl

Publication:
"The open motion planning library", Sucan et al., RAM, 2012

PlannerArena

Online at:
https://plannerarena.org

Publication:
"Benchmarking motion planning algorithms: An extensible infrastructure for analysis and visualization," Moll et al., RAM, 2015

HyperPlan

Public Repository:
https://github.com/Kavrakilab/hyperplan

Publication:
"HyperPlan: A Framework for Motion Planning Algorithm Selection and Parameter Optimization," Moll et al., IROS, 2021

Robowflex

Public Repository:
https://github.com/KavrakiLab/robowflex

Publication:
"Robowflex: Robot Motion Planning with MoveIt Made Easy," Kingston et al., Arxiv 2021

MotionBenchMaker

Public Repository:
https://github.com/Kavrakilab/motion_bench_maker

Publication:
"MotionBenchMaker: A Tool to Generate and Benchmark Motion Planning Datasets," Chamzas et al., RAL, 2022

Some Sampling-Based Motion Planners in OMPL

Multi-query Planners
- **PRM** (Kavraki et al 1996)
- **LazyPRM** (Bohlin & Kavraki 2000)
- **QMP** (Orthey et al. 2018)

Single-Query Planners
- **RRT** (Kuffner & Lavalle 2000)
- **RRTConnect** (Kuffner & Lavalle 2000)
- **EST** (Hsu et al 1999)
- **SBL** (Sancez & Latombe 2001)
- **KPIECE** (Sucan & Kavraki 2008)
- **STRIDE (**Gibson et al. 2013)
- **PDST** (Ladd & Kavraki 2005)
- **QRT** (Orthey & Toussaint 2019)

Kinodynamic Planners
- **RRT** (Kuffner & Lavalle 2000)
- **EST** (Hsu et al 1999)
- **KPIECE** (Sucan & Kavraki 2008)
- **Syclop** (Plaku et al. 2010)
- **PDST** (Ladd & Kavraki 2005)
- **Sparse Stable RRT** (Li et al. 2016)

Constrained Planners
- **CBiRRT2** (Berenson et al. 2011)
- **AtlasRRT** (Jaillet & Porta 2013)
- **Tangent Bundle RRT** (Kim et al. 2016)
- **General Manifold Constraints** (Kingston et al. 2019)

Optimizing Planners
- **CFOREST** (Otte & Correl 2013)
- **Anytime Path Shortening** (Luna et al. 2013)
- **RRT*** (Karamazon & Frazzoli 2011)
- **PRM*** (Karaman & Frazzoli 2011)
- **SPARS** (Dobson et al. 2012)
- **SPARS2** (Dobson & Bekris 2013)
- **Transition-Based RRT** (Jaillet et al. 2000)
- **LBTRRT** (Salzman & Halperin 2013)
- **Sparse Stable RRT** (Li et al. 2016)
- **Vector Field RRT** (Ko et al. 2014)
- **ST-RRT*** (Grothe et al. 2022)
- **RRT#** (Arslan & Tsiotras 2015)
- **RRTx** (Otte & Frazzoli 2015)
- **InformedRRT*** (Gammel et al. 2014)
- **FMT*** (Janson et al. 2015)
- **BFMT*** (Starek et al. 2015)
- **BIT*** (Gammel et al. 2015)
- **AIT*** (Strub and Gammel 2022)
- **EIT*** (Strub and Gammel 2022)

We thank NSF
for their support

Thank you for your attention!