# Platform-Independent Benchmarks for Task and Motion Planning

Fabien Lagriffoul, Neil T. Dantam, Caelan Garrett, Aliakbar Akbari, Siddharth Srivastava, and Lydia E. Kavraki

***Abstract*—We present the first platform-independent evaluation method for task and motion planning (TAMP). Previously point, various problems have been used to test individual planners for specific aspects of TAMP. However, no common set of metrics, formats, and problems have been accepted by the community. We propose a set of benchmark problems covering the challenging aspects of TAMP and a planner-independent specification format for these problems. Our objective is to better evaluate and compare TAMP planners, foster communication, and progress within the field, and lay a foundation to better understand this class of planning problems.**

***Index Terms*—Performance evaluation and benchmarking, task planning, manipulation planning.**

## I. Introduction

EVERYDAY activities require reasoning about both high-level task actions and specific geometric motions. Systems that perform both task planning and motion planning have existed for a long time, e.g., the pioneering robot Shakey [1]. These systems were traditionally rooted in the classical *sense-plan-act* paradigm, which separated the planning of logical task actions from the execution of specific motions. However, the increasing complexity of robots and their capacity to manipulate the environment has uncovered a new class of problems, which cannot be solved by considering task actions and geometric motions in isolation. Many activities feature *dependencies* between logical and geometrical levels. Deciding *which* actions to do—and in which order—is closely linked to *how* these actions can be physically performed. Thus, it requires planning techniques that combine both task planning and motion planning.

Combined task and motion planning is an active research area with work proceeding in the robotics, AI, and formal methods communities. Though the broad aims of these works are similar, the focus spans many related areas: planar navigation [2], rearranging objects [3], dynamics in manipulation [4], humanoid robots [5], achieving optimality [6]. The variation in focus, assumptions, and scenarios presents challenges when attempting to directly compare planners and measure the forward progress of the field. *This letter presents a set of criteria, metrics, and scenarios for the evaluation, comparison, and benchmarking of Task-Motion planners.*

The evaluation approach in this letter is a collaborative development following TAMP workshops at RSS 2016 and 2017. We have produced a common set of benchmarks[1] that are independent of any of our specific planning systems. Beyond the initial comparison of planners, we believe that a common set of benchmarks—and ultimately a planning competition—is crucial to promote development and *measure research progress*, as demonstrated by the success of the International Planning Competition (IPC) [7], the Satisfiability Modulo Theories Competition (SMT-COMP) [8], or robotics competitions such as RoboCup [9] and RoCKIn [10]. Through the present work, we anticipate that other researchers will join our initiative to improve and refine this evaluation method or even develop a superior successor. The fundamental prerequisite for this benchmarking approaches is community consensus on the choice of benchmark problems and methods. This letter is a first step in this direction.

The letter is organized as follows: Section II presents prior work on TAMP. Section III gives an overview of TAMP and discusses the assumptions made in this letter. Based on these assumptions, we formally define TAMP as considered for this work in Section IV. Section V explains the data and specification formats we adopt for TAMP problems. Section VI introduces the benchmarks problems and discusses how they were developed. Section VII offers practical details about using the benchmark data and evaluation metrics. Finally, Section VIII provides a summary and perspective on future work.

Manuscript received February 24, 2018; accepted June 28, 2018. Date of publication July 16, 2018; date of current version August 8, 2018. This letter was recommended for publication by Associate Editor O. Stasse and Editor N. Tsagarakis upon evaluation of the reviewers' comments. This work was supported in part by the Swedish Knowledge Foundation (KKS) project "Semantic Robots", NSF Grants IIS 1317849 and CCF-1514372 to the Kavraki Lab at Rice University, and in part by the Spanish Government through the project DPI2016-80077-R. The work of A. Akbari was supported by the Spanish Government through the Grant FPI 2015. *(Corresponding author: Fabien Lagriffoul.)*

F. Lagriffoul is with the Centre for Applied Autonomous Sensor Systems Lab, Örebro University, Örebro SE-701 82, Sweden (e-mail: Fabien.Lagriffoul@oru.se).

N. T. Dantam is with the Colorado School of Mines, Golden, CO 80401 USA (e-mail: ndantam@mines.edu).

C. Garrett is with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: caelan@mit.edu).

A. Akbari is with the Universidad Politécnica de Catalunya, Barcelona 08034, Spain (e-mail: aliakbar.akbari@upc.edu).

S. Srivastava is with the Arizona State University, Tempe AZ 85281 USA (e-mail: siddharths@asu.edu).

L. E. Kavraki is with the Rice University, Houston, TX 77005 USA (e-mail: kavraki@rice.edu).

Digital Object Identifier 10.1109/LRA.2018.2856701

[1] http://tampbenchmark.aass.oru.se

## II. Prior Work on TAMP

The first work combining logical and geometric search spaces originates from both the robotics and AI communities. Cambon *et al.* developed aSyMov [11] in 2004, a motion planner for multiple robots using AI planning as a heuristic for probabilistic roadmap search. Likewise in the AI planning community, Dornhege *et al.* introduce *semantic attachments* in 2009 in order to account for the geometric side effects of logical actions [12]. Navigation Among Movable Obstacles (NAMO) [13] relates to TAMP in that it combines motion planning together with discrete choices about obstacles to move. However, NAMO focuses on a specific type of discrete action (moving obstacles) in contrast to general task planning in TAMP.

TAMP has become an active field of research in both AI and Robotics communities, reflected by the emergence of dedicated workshops, conference tracks, and journal special issues: AAAI 2010 (workshop on Bridging the gap between Task and Motion Planning), ICAPS (PlanRob workshop since 2012, Robotic Track since 2015), RSS since 2013 (workshop on Task and Motion Planning), IROS 2013–2015 (AI-based Robotics, AI and Robotics, Task Planning for Intelligent Robots), AI Journal 2014 (special issue on AI and Robotics), and the International Journal of Robotics Research's current call for a special issue on Task and Motion Planning.

Planning systems combining both types of planning have flourished [4], [14]–[23], together with their own planner-specific evaluations, yet no common benchmarks have so far been established. These systems are evaluated on subsets of possible problems which do not cover all the features of TAMP, therefore leading to (over)specialized planners [24]. The topic of benchmarks for TAMP is recurrently proposed in dedicated workshops, and we hope through this letter to finally address the issue.

## III. Overview of TAMP

TAMP combines logical and geometric reasoning. Robots need logical reasoning to determine which actions are needed to achieve a given goal, and they need geometric reasoning to know if and how these actions can be physically performed. TAMP addresses this issue by combining discrete task decisions about objects and actions with continuous motion decisions about paths.

Typical algorithms for independent task planning and motion planning are fundamentally different. Task planning finds a discrete sequence of actions to transition from a given start state to a desired goal condition, typically using heuristic search or constraint satisfaction. Geometric motion planning finds a collision-free path from a given start configuration to a desired goal, typically using sampling or optimization-based methods. TAMP combines these two planning domains and addresses the interaction between them.

The range of problems which may be considered "TAMP" is large, spanning geometric vs. dynamic, fully vs. partially observable, deterministic vs. non-deterministic, and single-agent vs. adversarial vs. collaborative cases. For the scope of this work, we have necessarily restricted the focus to produce a workable benchmark set. We hope that further evaluation and developments will lead to standardized benchmarks for broader problem classes in the future.

### A. Assumptions

This benchmark set addresses the *geometric, fully-observable, deterministic, single-agent* subset of TAMP. We focus on the computational complexity of searching coupled logical and geometric spaces, in contrast to other areas of robotics which may primarily focus on uncertainty and partial observability. Hence, we make the following assumptions:

- **Geometric:** Motion planning over positions only is sufficient. Objects that are grasped or placed are kinematically coupled with the parent object and do not move or slide.
- **Fully Observable:** The initial state is entirely known geometrically (positions, meshes, configurations), semantically (e.g., movability of objects, placement locations).
- **Deterministic:** State is only changed by the planned actions. Robot motions and object grasp/release operations exactly follow the output of the path planner.

### B. Specification of TAMP Problems

*1) Task Requirements:* At a high level, task planners search through a discrete transition system (see Def. 1). We need a specification format that is sufficiently *expressive* and *compact* to represent our domains of interest. The Planning Domain Definition Language (PDDL) [25], [26] is a task modeling language with wide support due to its use in the International Planning Competition [7]. PDDL's status as a de facto standard makes it a natural choice to exchange task domains for this benchmark set. Though the problems are specified in PDDL (see Section V-A), we anticipate that task-motion planners will also use other representations internally, e.g., answer set programming (ASP) [27], SMTLib [28], or temporal logics, based on specific implementation decisions. We discuss the details of the task specification format in Section V-A.

*2) Motion Requirements:* Abstractly, motion planners find paths through a (typically continuous) configuration space. Thus, we need to specify the configuration space for the motion planner. For this initial benchmark set, we focus on geometric motion planning and leave dynamic or physics-based planning [4], [29], [30] to future benchmark sets. The geometric case is sufficient for many manipulation scenarios. The configuration space for the motion planner is based on the robot's kinematics, i.e., joints, and the positions of the rigid bodies in the environment. There are many equivalent formats to specify robot kinematics and rigid body geometry. We discuss the details of the motion specification format in Section V-B.

## IV. TAMP Problem Definition

We formulate the TAMP problem starting from the conventional formulations of task planning [31] and motion planning [32], [33].

### A. Task Planning

Task planning finds a sequence of discrete or symbolic actions from an initial state to a desired goal state. While task

planning covers a wide range of problems, all notations and representations correspond to a *state-transition-system* [34]:

*Definition 1 (Task State-Transition-System):* A task domain is the tuple $\Sigma = (S, A, \gamma, s_0, S_g)$ where,

- $S$ is a finite set of states
- $A$ is a finite set of actions
- $\gamma : S \times A \rightarrow S$ is a deterministic state-transition function, thus gives one state when applicable, which we denote by $\gamma(s, a) = s'$.
- $s_0 \in S$ is the start state
- $S_g \subseteq S$ is the set of goal states

*Definition 2 (Task Plan):* A task plan for domain $\Sigma = (S, A, \gamma, s_0, g)$ is the sequence $\langle a_0, a_1, \ldots, a_n \rangle$, where each $a_i \in A$, for $0 \leq i < n,\ s_{i+1} = \gamma(s_i, a_i)$, and $s_n \in g$. That is, the task plan is a string in the language of $\Sigma$.

Since for even small task planning problems, it is computationally infeasible to explicitly represent all states $S$, one must instead use various compact, symbolic representations, e.g., the Planning Domain Definition Language (PDDL) [25], Linear Temporal Logic (LTL) [35], Answer Set Programming (ASP) [27], $\mathcal{C}+$ [36], etc.

### B. Motion Planning

Abstractly, motion planning finds valid paths through a *configuration space* $\mathcal{C}$ from a given start to a given goal state. The *free configuration space*, $\mathcal{C}_{\text{free}} \subseteq \mathcal{C}$, is the space of configurations, e.g., configurations where the robot does not collide with objects or itself.

*Definition 3 (Motion Plan):* Given free configuration space $\mathcal{C}_{\text{free}}$, initial configuration $q_I \in \mathcal{C}_{\text{free}}$, and goal configurations $G \subseteq \mathcal{C}_{\text{free}}$, a motion plan is defined as either:

1) A sequence $Q = \langle q_0, \ldots, q_n \rangle$ where $q_0 = q_I$ is the start configuration, $q_n \in G$ is a goal configuration, each $q_i \in \mathcal{C}_{\text{free}}$ is valid, and subsequent configurations are nearby $\|q_{i+1} - q_i\| \leq \epsilon$;
2) Or a continuous trajectory $\tau : [0, 1] \rightarrow \mathcal{C}_{\text{free}}$ such that $\tau(0) = q_I$ and $\tau(1) \in G$.

Typically, we will conduct motion planing in planar or 3-dimensional worlds— $\mathcal{W} = \mathbb{R}^2$ or $\mathcal{W} = \mathbb{R}^3$—with obstacle region $\mathcal{O}$, and one or more multi-jointed robots. A configuration $q \in \mathcal{C}$ corresponds to joint position vector for the robot, and a free configuration $q \in \mathcal{C}_{\text{free}}$ is a joint position vector not in collision. Widely-used software packages model the configuration space of robot manipulators using kinematic trees or scene graphs [37]–[39] of local coordinate frames and attached rigid-body geometry (i.e., meshes), connected by the robots' joints. Explicit representations of the free configuration space $\mathcal{C}_{\text{free}}$ are generally infeasible to produce, so we instead use "black-box" collision checkers [40] to test the validity (freedom-from-collision) of configurations during planning.

### C. Task-Motion Planning

The interaction between task planning and motion planning is fundamental to TAMP. However, different planning approaches have formalized task-motion interaction differently. Thus, we first informally discuss the necessary relations between task planning and motion planning. Then, we summarize different specific formalizations. Finally, we present a minimal interaction specification for the benchmark set.

TAMP requires that we establish relationships between (1) states and configurations and (2) actions and motion plans. These relations ensure *consistence* between symbolic and geometric domains. Consistence implies, for instance, that the initial state $s_0 \in S$ correspond to the initial configuration $q_0 \in \mathcal{C}$. Similarly, each action $a \in A$ in the task plan can only occur if there exists a corresponding feasible motion plan $\tau_a \in \mathcal{C}_{\text{free}}$. Actions that change the free configuration space—e.g., grasping an object with the robot's gripper—must be reflected at the motion planning level, e.g., with an additional fixed joint between object and gripper. Finally, symbolic-geometric consistence implies that during execution of a motion plan, the world does not undergo uncontrolled state transitions, i.e., given the state transition $\gamma(s, a) = s'$, the configurations traversed by the motion plan cannot be mapped to a state other than $s$ or $s'$. Based on these necessary relations, we now summarize different specific approaches to task-motion interaction.

Various methods and formalizations of task-motion interaction are used by different planning systems. The *domain semantics* approach introduces a pair of functions to define task-motion interaction in terms of *abstraction* and *refinement* [41], [42]. The *abstraction* function maps from a scene graph to a task state, and the *refinement* function maps from an action to a motion planning goal and an updated scene graph. Similar to the aforementioned refinement, *Semantic attachments* [12] connect PDDL-type actions to external geometric procedures, and the *Planner-Independent Interface Layer* [19] implements a symbolic-to-geometric mapping. Pre-computing *reachability maps* is another approach to ensure the semantics of a task domain or a logic program to reflect geometric constraints [14], [16], [43]. The *sample-based* approach [44] introduces conditional samplers to generate configurations corresponding to task states or actions. For example, a conditional sampler for a grasp action would generate configurations based on inverse kinematics to put the manipulator in a grasping configuration.

For the scope of this letter, we adopt a simplified model of task-motion interactions with the aim of maximizing compatibility with various planning systems. We use two abstract mapping functions:

- $\phi : S \rightarrow 2^C$ which maps states to configurations;
- $\xi : A \rightarrow 2^C$ which maps actions to motion plans.

As an example, if task state $s$ indicates "the robot is in the kitchen," $\phi(s)$ encompasses *all* the configurations such that the robot is located in the kitchen, grasping or not all available objects in any possible ways, the remaining objects lying in all possible poses. $\phi$ and $\xi$ are further described in Section V-C when we present a concrete implementation of task-motion interactions in the context of manipulation problems.

Based on this minimal specification of task-motion interaction, we now define the TAMP problem.

*Definition 4 (Task-Motion Planning):* Given a tuple $(\mathcal{C}, \Sigma, \phi, \xi, q_0)$, the TAMP problem is to find a sequence of actions $\langle a_0, a_1, \ldots, a_{n-1} \rangle$ following a sequence of task states

```
(:action move
        :parameters (?disc ?from ?to)
        :precondition (and (smaller ?to ?disc)
                           (on ?disc ?from)
                           (clear ?disc) (clear ?to))
        :effect (and (clear ?from)
                     (on ?disc ?to)
                     (not (on ?disc ?from))
                     (not (clear ?to))))
```

(a)

```
(:init
 (smaller peg1 disc1) ... (smaller peg3 disc3)
 (smaller disc2 disc1)
 (smaller disc3 disc1)
 (smaller disc3 disc2
          (clear peg2) (clear peg3) (clear dic1)
          (on disc3 peg1) (on disc2 disc3) (on disc1 disc2))
 (:goal
  (and (on disc3 peg3) (on disc2 disc3) (on disc1 disc2)))
```

(b)

Fig. 1. Example PDDL task specification for the Towers of Hanoi problem (see Section VI-A). (a) Represents the transition function $\gamma$ of the task language (see Def. 1) using PDDL actions with preconditions and effects. (b) Represents the start and goal states of the task language using symbolic expressions.

$\langle s_0, s_1, \ldots, s_n \rangle$ such that

$$s_n \in S_g \text{ and} \tag{1}$$

$$s_{i+1} = \gamma(s_i, a_i) \tag{2}$$

and to find a sequence of motion plans $\langle \tau_0, \tau_1, \ldots, \tau_{n-1} \rangle$ such that $\forall i = 0 \ldots n-1$:

$$\tau_i(0) \in \phi(s_i) \text{ and } \tau_i(1) \in \phi(s_{i+1}) \tag{3}$$

$$\tau_i \in \xi(a_i), \text{ and} \tag{4}$$

$$\tau_i(1) = \tau_{i+1}(0) \tag{5}$$

## V. Specification of Benchmark Problems

The data and formats to specify benchmark problems are a necessary part of any planner-independent test suite for TAMP. Thanks to existing benchmarks, competitions [7], and software packages [39], there are already many standard formats and languages for task planning and motion planning in isolation. Thus, we adopt the standard and conventional formats for the task domain (see Section IV-A) and the motion domain (Section IV-B). For task-motion interaction (Section IV-C) at the current stage, we provide high-level semantic information on grasp points, placement locations, etc. with the hope of stimulating further testing and development leading to an eventual standard format.

### A. Task Specification

The de facto standard format for task domains is the Planning Domain Definition Language (PDDL) [26] developed for the International Planning Competition [7]. PDDL defines the task language in terms of actions with symbolic (Boolean expression) preconditions and effects, an initial state, and a symbolic expression for the set of goal states. A transition in the task language corresponds to taking a PDDL action, where the PDDL precondition holds in the predecessor state and the PDDL effect sets the successor state. PDDL compactly represents large state spaces using symbolic expressions for sets of states. Fig. 1 shows an example PDDL specification for the classic Towers of Hanoi problem, detailed in Section VI-A.

While we adopt PDDL for the benchmark set as a standard interchange format for transitions sytems, this choice does not preclude use of other representations, e.g., ASP, LTL, within planners. The choice of modeling language may, however, influence the focus of planners [45], and we hope further use of the benchmarks will yield refinements to the task specification format.

### B. Motion Specification

The essentially universal model for robot manipulators is the kinematic tree or scene graph with attached rigid body geometry (meshes) [37]–[39], [42]. However, there are various alternative formats to specify the tree structure and the mesh data. We briefly summarize some major formats and explain the choices for the benchmark set aimed at maximizing compatibility.

*1) Kinematics:* Historically, Denavit-Hartenberg (DH) parameters [46] were a common approach to specify manipulator kinematics, though they did not address the meshes necessary for collision detection. The ROS Universal Robot Definition Format (URDF) [47] is a currently popular representation for robot manipulators with widespread support from robot vendors and the ability to reference external mesh files. Due to the wide availability of URDF models for many robots, we specify the benchmark problems using URDF.

*2) Geometry:* Numerous formats for mesh data have been developed by the animation and computer-aided design (CAD) communities. One attempt at a universal interchange format is the XML-based COLLADA [48]. However, the COLLADA specification is difficult to implement and software support and compatibility is poor. The Wavefront Object (OBJ) format is a lightweight representation for mesh data that is human-readable and easy to parse. Wavefront OBJ is widely supported among animation and CAD software packages. Thus, we specify meshes for the benchmark set using OBJ.

### C. Specification of Task-Motion Interaction

TAMP planners use their own symbolic-geometric mappings, hence there is currently no standard language to specify task-motion interaction. This stems from the fact that (1) at the task level, domains can be modeled at arbitrary levels of abstraction, and (2) at the motion level, there are various ways of implementing motion primitives. Consider for instance a planner using the following motion primitive for grasping: "*reach X with the manipulator, close the gripper, lift X off the table by 5 cm,*" while another planner uses: "*reach X with full-body motion, close the gripper*." Comparing both planners would not be fair because they do not afford the same possibilities of action in the first place, therefore they cannot reach the same solution space.

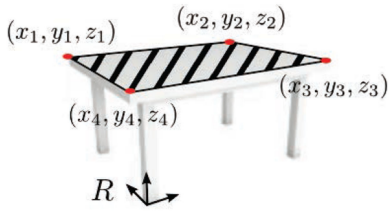


Fig. 2. Rectangular SSSP for a table represented by the list $((x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3), (x_4, y_4, z_4))$.

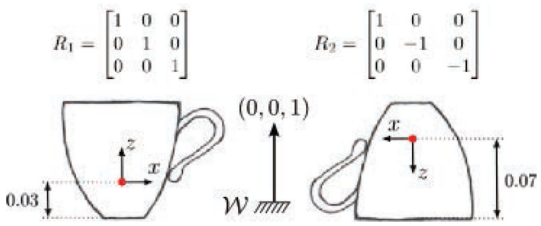


Fig. 3. Two SOPs (upright and upside down) for a cup represented by the set $\{(R_1, (0, 0, 1), 0.03), (R_2, (0, 0, 1), 0.07)\}$.

To promote fair comparison of planners, we provide simplified symbolic-geometric mappings $\phi$ and $\xi$ (see Section IV-C) for manipulation scenarios. This semantic information is based on both the properties of the robot and environment. The existing Semantic Robot Definition Format (SRDF) provides some of necessary information but does not fully cover the needs of Task-Motion interaction. We provide the data relating task actions and motions as XML files[2] included with the benchmarks.

*1) State-Configuration Mapping ($\phi$):* We make the following assumptions which are restrictive, but allow us to specify $\phi$ unambiguously and make sure that different planners actually address the same problems:

- *Surfaces Supporting Stable Placement (SSSP)*: When moved, objects can only be placed at these regions. An SSSP is a flat polygon, segment, or point attached to an object (see Fig. 2).
- *Stable Object Poses (SOP)*: At rest, objects have to lie in one of these predefined poses. An SOP is represented by a rotation matrix, an axis of rotation (both in world frame) and a distance (to a SSSP) (see Fig. 3).

For each object, an *optional* set of grasps is provided, i.e., the planner may use it or not. A discrete grasp is represented by a grasp frame (in the reference frame of the object), and a continuum of grasps is represented by a single grasp frame plus an axis of rotation (see Fig. 4).

*2) Action-Motion Plan Mapping ($\xi$):* Specifying how symbolic actions relate to motion plans is partly done in definition 4, which imposes start and goal configurations (3) and continuity constraints (5). Only motion primitives needs to be specified. From our experience, there are as many motion primitives as there are system designers. In order to keep the benchmarks accessible to a majority of planners, we only specify which joints are allowed to be actuated for a given action. No further details

[2]http://tampbenchmark.aass.oru.se/index.php?title=XML_format_description

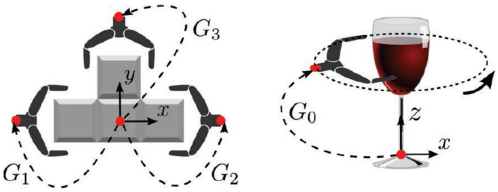


Fig. 4. A discrete grasp set : $\{G_1, G_2, G_3\}$ (left) and a continuous grasp set $(G_o, (0, 0, 1))$ (right).

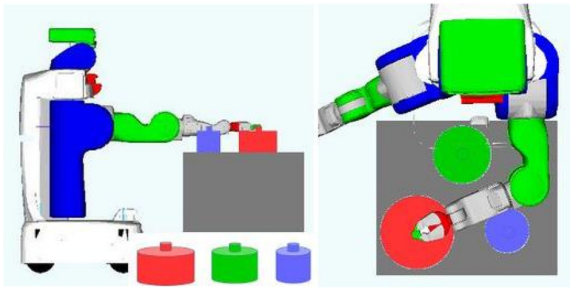

Fig. 5. The *Towers of Hanoi* benchmark.

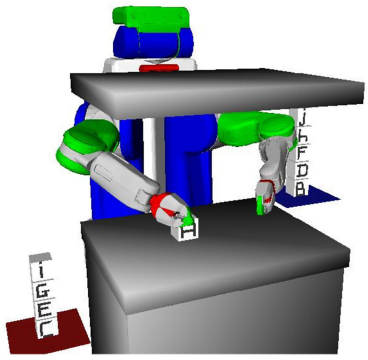

Fig. 6. The *Blocks World* benchmark.

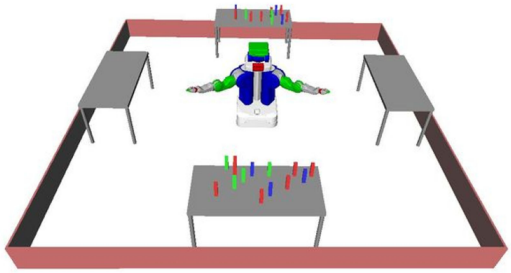

Fig. 7. The *Sort Clutter* benchmark.

about motion planning are specified, i.e., we consider the *maximum* number of degrees of freedom as the common ground. Note that it is perfectly valid to actuate a subset of the proposed joints. This leaves freedom for the users in the design of their motion primitives. On the other hand, it may be the case that a benchmark requires, e.g., the manipulators to be actuated separately (because the problem is more challenging that way). This would imply for some users (e.g., using only dual-arm motion

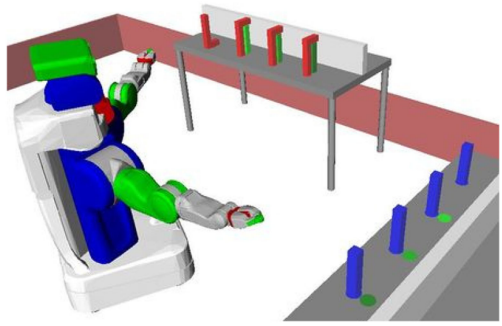

Fig. 8. The *Non-Monotonic* benchmark.

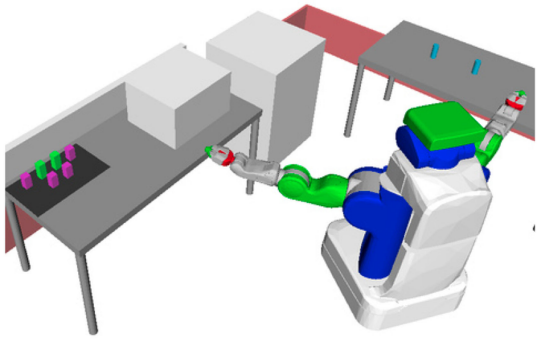

Fig. 9. The *Kitchen* benchmark.

TABLE I
CRITERIA EVALUATED BY EACH BENCHMARK PROBLEM

| Criteria | Pb. 1 | Pb. 2 | Pb. 3 | Pb. 4 | Pb. 5 |
|---|---|---|---|---|---|
| Infeasible task actions | ✓ | ✓ | ✓ | ✓ | ✓ |
| Large task spaces | ✓ | ✓ | ✓ | | |
| Motion/Task Trade-off | | ✓ | | | |
| Non-monotonicity | | | | ✓ | ✓ |
| Non-geometric actions | | | | | ✓ |

planning) to implement new motion primitives to comply with the specifications. This information is given in the XML files[4] aforementioned.

## VI. BENCHMARKS

This section examines five benchmark problems selected to evaluate planners based on the criteria in Table I. These criteria feature properties which make TAMP problems computationally difficult, based on scenarios from TAMP literature, experience with empirically difficult TAMP problems, and theoretical studies on difficult TAMP criteria [13], [49]. We consider the following criteria to evaluate TAMP in the fully-observable, deterministic case:

- **Infeasible task actions:** Some task actions are not possible, i.e, no corresponding motion plan exists. Possible causes of infeasibility include blocking objects and kinematic limits of the robot.
- **Large task spaces:** The underlying task planning problem requires substantial search effort.
- **Motion/Task Trade-off:** The problem can be solved with fewer steps if grasps and placements are carefully chosen.
- **Non-monotonicity:** Some objects need to be moved more than once for achieving the goal.
- **Non-geometric actions:** The problem involves actions which change discrete state $\Sigma$ but not configurations $\mathcal{C}$.

### A. Problem 1: Towers of Hanoi

We extend the classic Tower of Hanoi problem to robot manipulation. The base of the robot is fixed. The rods are set in a triangular fashion and the discs are very thick, so that stacking them on a rod may temporarily create an obstacle and prevent from picking/placing discs on other rods.

This problem evaluates the *large task space* and *infeasible task actions* criteria. Although the rules are simple, the optimal solution plan for $n$ discs contains $2^n - 1$ steps, without considering geometry (large task space). The rules of the game require certain intermediate states, many of which are geometrically infeasible (infeasible task actions) if the chosen order creates occlusions.

### B. Problem 2: Blocks World

Another classical task planning problem, in which the goal is to stack the blocks in alphabetical order anywhere it is possible. The base is fixed, only top-grasps can be used, and hand-over is not allowed.

This problem evaluates the *infeasible task actions*, *large task space*, and *motion/task trade-off* criteria. An obstacle is hovering over the table, such that the grippers would collide with it if more than two blocks are stacked on the table (infeasible task actions). Both initial piles need to be un-stacked somewhere and re-stacked on one of the trays. During this process, half of the blocks need to transit from one tray to another, through the table. This requires many actions (large task space). The region on the table where blocks transit has to be reachable by both arms, therefore its size is limited. One must choose between few steps and cluttered table, or many steps and uncluttered table (motion/task trade-off).

### C. Problem 3: Sort clutter

The goal constraints are that all $N$ blue blocks must be on the left table and all $N$ green blocks must be on the right table. There are also $2N$ red blocks acting as obstacles for reaching blue and green blocks.

This problem evaluates the *infeasible task actions* and *large task space* criteria. The close proximity of the blocks forces the planner to carefully order its operations, as well as to move red blocks out of the way without creating new occlusions (infeasible task actions). Solving the problem requires to move many objects, sometimes multiple times (large task space).

### D. Problem 4: Non-Monotonic

The robot must move the green blocks from the left table to a corresponding position on the right table. In the goal state, blue and cyan blocks have to be in their initial poses.

This problem evaluates the *infeasible task actions* and *non-monotonicity* critiera. Both the initial and goal poses are blocked by four blue and cyan blocks respectively (infeasible task actions). The goal condition of blue and cyan blocks requires to temporarily move them away and bring them back later on (non-monotonicity) in order to solve the problem.

### E. Problem 5: Kitchen

The robot must "prepare a meal" by cleaning two glasses (blue blocks), cooking two cabbages (green blocks), and setting the table. Objects can be cleaned when placed on the dishwasher and cooked when placed on the microwave. An object must be cleaned before it can be cooked. Finally, the radishes (pink blocks) initially obstruct the cabbages on the shelf forcing the robot to move them. But, to maintain a tidy kitchen, the robot must also return them to their initial poses.

This problem evaluates the *infeasible task actions*, *non-monotonicity*, and *non-geometric actions* criteria. A `cook` action is non-monotonic and both `cook` (via the stove) and `clean` (via the dishwasher) are non-geometric from the robot's perspective. Reaching target objects may also require removing blocking objects.

## VII. Using the Benchmark Set

The current benchmark set is independent of specific planners and specific robots to maximize portability to different platforms and support physical testing on available hardware. We have, however, used the PR2 as an example since it is a commonly-used robot. We anticipate that the problems will port to other robots, though some cases may require slight modification. For example, the *Blocks World* benchmark (see Section VI-B) may require different placements of the trays, or a different position of the hovering obstacle, because reachability and occlusions may differ for some other robots.

Each problem is available in different scales. For instance, the *Towers of Hanoi* problem is proposed with 3, 4, 5, and 6 discs. For each scale, there are $N$ random variations of the initial geometric state, over which the average values of the metrics are computed.

### A. Files

The proposed benchmarks are available online at the following address: http://tampbenchmark.aass.oru.se/index.php?title=Problems

For each problem, we provide following data to define the task domain, motion domain, and task motion interaction:

1) A PDDL file defining the task domain
2) An archive containing all scene meshes in Wavefront OBJ format
3) An XML file describing:
   a) the initial pose of all objects;
   b) the initial configuration of the robot(s);
   c) the initial kinematic coupling between objects;
   d) which objects are movable;
   e) the Surfaces Supporting Stable Placement (SSSP);
   f) the Stable Object Poses (SOP);
   g) Grasp Sets (optional).

### B. Metrics

We propose to quantitatively compare TAMP systems based on metrics that evaluate performance of the planner itself and quality of the computed plans. Planner performance can be measured in terms of average planning times, overall success rate, or success rate within a time bound. For the current benchmarks, we propose measuring plan quality in terms of length of the computed plan, both for the number of actions in the task plan and total length of the motion plans. Multiple metrics that cover efficiency, successful instances, and plan quality will measure the trade-offs between different TAMP approaches with different focus, for example expected efficiency vs. achieving probabilistic completeness.

## VIII. Conclusion

In this letter, we have proposed a principled approach to evaluate and compare TAMP systems. Though work in TAMP has proceeded for more than a decade, it has remained until now difficult to directly compare different planners. We have formally define the TAMP problem under full observability and determinism assumptions, presented a way to unambiguously specify TAMP problems, and proposed five benchmark problems selected to cover the known challenges of TAMP. Finally, we provide a web page (http://tampbenchmark.aass.oru.se) to download the necessary data for the benchmark problems.

Our continuing step is to organize a workshop to use and discuss this benchmark set. Based on feedback from the community, the proposed problems and specification formats may be revised and improved in several possible ways. Rather than individual problem instances, we could produce problem generators with tunable hardness parameters, based on community evaluation of the criteria in Table I. One could extend the class of problems, e.g., to partially observable, non-deterministic cases. Another possible direction is to organize a TAMP competition, which would require a common output format for plans and an automatic plan validator. We will continue development and dissemination of this benchmark set to promote direct comparison among TAMP systems.

## References

[1] N. J. Nilsson, "Shakey the robot," AI Center, SRI Int., Menlo Park, CA, USA, Tech. Rep. 323, Apr. 1984.
[2] M. Stilman and J. J. Kuffner, "Navigation among movable obstacles: Real-time reasoning in complex environments," *Int. J. Humanoid Robot.*, vol. 2, no. 4, pp. 479–503, 2005.
[3] A. Krontiris and K. Bekris, "Dealing with difficult instances of object rearrangement," in *Proc. Robot., Sci. Syst.*, Rome, Italy, Jul. 2015, [Online]. Available: http://www.roboticsproceedings.org/
[4] A. Akbari, Muhayyudin, and J. Rosell, "Task planning using physics-based heuristics on manipulation actions," in *Proc. 21st Int. Conf. Emerg. Technol. Factory Automat.*, 2016, pp. 1–8.
[5] L. Karlsson, J. Bidot, F. Lagriffoul, A. Saffiotti, U. Hillenbrand, and F. Schmidt, "Combining task and path planning for a humanoid two-arm robotic system," in *Proc. TAMPRA: ICAPS Workshop Combining Task Motion Planning Real-World Appl.*, 2012, pp. 5–12.

[6] W. Vega-Brown and N. Roy, "Asymptotically optimal planning under piecewise-analytic constraints," in *Proc. Workshop Algorithmic Foundations Robot.*, 2016. [Online]. Available: http://wafr2016.berkeley.edu/papers/

[7] M. Vallati *et al.*, "The 2014 international planning competition: Progress and trends," *AI Mag.*, vol. 36, no. 3, pp. 90–98, 2015.

[8] C. Barrett, M. Deters, L. de Moura, A. Oliveras, and A. Stump, "6 years of SMT-COMP," *J. Automated Reason.*, vol. 50, no. 3, pp. 243–277, Mar. 2013.

[9] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa, "Robocup: The robot world cup initiative," in *Proc. First Int. Conf. Auton. Agents*, 1997, pp. 340–347.

[10] A. Saffiotti and T. v. d. Zant, "Foreword: The impact of RoCKIn on robotics," in *RoCKIn—Benchmarking Through Robot Competitions*. Rijeka, Croatia: InTech, 2017, ch. 1.

[11] S. Cambon, F. Gravot, and R. Alami, "A robot task planner that merges symbolic and geometric reasoning," in *Proc. Eur. Conf. Artif. Intell.*, 2004, pp. 895–899.

[12] C. Dornhege, P. Eyerich, T. Keller, S. Trüg, M. Brenner, and B. Nebel, "Semantic attachments for domain-independent planning systems," in *Proc. Int. Conf. Automat. Planning Scheduling*, 2009, pp. 114–121.

[13] M. Stilman, J.-U. Schamburek, J. Kuffner, and T. Asfour, "Manipulation planning among movable obstacles," in *Proc. IEEE Int. Conf. Robot. Automat.*, Apr. 2007, pp. 3327–3332.

[14] J. Choi and E. Amir, "Combining planning and motion planning," in *Proc. Int. Conf. Robot. Automat.*, 2009, pp. 238–244.

[15] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *Proc. Int. Conf. Robot. Automat.*, 2011, pp. 1470–1477.

[16] E. Plaku, "Path planning with probabilistic roadmaps and linear temporal logic," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 2269–2275.

[17] L. de Silva, A. K. Pandey, and R. Alami, "An interface for interleaved symbolic-geometric planning and backtracking." in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 232–239.

[18] R. Dearden and C. Burbridge, "An approach for efficient planning of robotic manipulation tasks," in *Proc. Int. Conf. Automat. Planning Scheduling*, 2013, pp. 55–63.

[19] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 639–646.

[20] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling, "Heuristic search for task and motion planning," in *Proc. Workshop Planning Robot.*, 2014, pp. 148–156.

[21] E. Erdem, K. Haspalamutgil, C. Palaz, V. Patoglu, and T. Uras, "Combining high-level causal reasoning with low-level geometric reasoning and motion planning for robotic manipulation," in *Proc. Int. Conf. Robot. Automat.*, 2011, pp. 4575–4581.

[22] N. T. Dantam, Z. Kingston, S. Chaudhuri, and L. E. Kavraki, "Incremental task and motion planning: A constraint-based approach," in *Robot., Sci. Syst.*, 2016, [Online]. Available: http://www.roboticsproceedings.org/

[23] F. Lagriffoul and B. Andres, "Combining task and motion planning: A culprit detection problem," *Int. J. Robot. Res.*, vol. 35, no. 8, pp. 890–927, 2016.

[24] F. Lagriffoul, "On benchmarks for combined task and motion planning," in *Proc. RSS Workshop Combined Task Motion Planning*, 2016, [Online]. Available: http://www.roboticsproceedings.org/

[25] M. Ghallab *et al.*, "PDDL—The Planning Domain Definition Language," 1998.

[26] S. Edelkamp and J. Hoffmann, "PDDL2. 2: The language for the classical part of the 4th international planning competition," Univ. Freiburg, Freiburg im Breisgau, Germany, Tech. Rep. 195, 2004.

[27] V. Lifschitz, "Answer set programming and plan generation," *Artif. Intell.*, vol. 138, no. 1, pp. 39–54, 2002.

[28] C. Barrett, P. Fontaine, and C. Tinelli, "The SMT-LIB Standard: Version 2.5," Dept. Comput. Sci, Univ. Iowa, Iowa City, IA, USA, Tech. Rep., 2015, [Online]. Available: www.SMT-LIB.org

[29] J. Barry, K. Hsiao, L. P. Kaelbling, and T. Lozano-Perez, "Manipulation with multiple action types," in *Proc. Int. Symp. Exp. Robot.*, Jun. 2012, pp. 531–545.

[30] M. Toussaint, "Logic-geometric programming: An optimization-based approach to combined task and motion planning," in *Proc. Twenty-Fourth Int. Joint Conf. Artif. Intell.*, 2015, pp. 1930–1936.

[31] D. Nau, M. Ghallab, and P. Traverso, *Automated Planning: Theory & Practice*. San Francisco, CA, USA: Morgan Kaufmann, 2004.

[32] S. M. LaValle, *Planning Algorithms*. New York, NY, USA: Cambridge Univ. Press, 2006.

[33] I. A. Şucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *Robot. Automat. Mag.*, vol. 19, no. 4, pp. 72–82, 2012.

[34] T. Dean and M. Wellman, *Planning and Control*. San Mateo, CA, USA: Morgan Kaufmann, 1991.

[35] E. M. Clarke, Jr, O. Grumberg, and D. A. Peled, *Model Checking*. Cambridge, MA, USA: MIT Press, 1999.

[36] E. Giunchiglia, J. Lee, V. Lifschitz, N. McCain, and H. Turner, "Nonmonotonic causal theories," *Artif. Intell.*, vol. 153, no. 1, pp. 49–104, 2004.

[37] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon Univ., Robot. Inst., Pittsburgh PA, USA, Aug. 2010. [Online]. Available: http://www.programmingvision.com/rosen_diankov_thesis.pdf

[38] R. Smits, H. Bruyninckx, and E. Aertbeliën, "KDL: Kinematics and dynamics library," 2011. [Online]. Available: http://www.orocos.org/kdl

[39] I. A. Şucan and S. Chitta, "MoveIt!" 2015. [Online]. Available: http://moveit.ros.org

[40] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," in *Proc. Int. Conf. Robot. Automat.*, 2012, pp. 3859–3866.

[41] N. T. Dantam, Z. Kingston, S. Chaudhuri, and L. E. Kavraki, "An incremental constraint-based framework for task and motion planning," *Int. J. Robot. Res.*, 2018.

[42] N. T. Dantam, S. Chaudhuri, and L. E. Kavraki, "The task motion kit," *Robot. Automat. Mag.*, May 2018.

[43] C. R. Garrett, T. Lozano-Perez, and L. P. Kaelbling, "Backward-forward search for manipulation planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 6366–6373.

[44] C. Garrett, T. Lozano-Perez, and L. Kaelbling, "Sample-based methods for factored task and motion planning," in *Proc. Robot., Sci. Syst.*, Jul. 2017, [Online]. Available: http://www.roboticsproceedings.org/

[45] J. Rintanen, "Impact of modeling languages on the theory and practice in planning research," in *Proc. Twenty-Ninth AAAI Conf. Artif. Intell.*, 2015, pp. 4052–4056.

[46] R. S. Hartenberg and J. Denavit, *Kinematic Synthesis of Linkages*. New York, NY, USA: McGraw-Hill, 1964.

[47] Willow Garage, "URDF XML," 2013. [Online]. Available: http://wiki.ros.org/urdf/XML

[48] International Organization for Standardization, *Industrial Automation Systems and Integration – COLLADA Digital Asset Schema Specification for 3D Visualization of Industrial Data*, 1st ed., ISO/PAS, 2012. [Online]. Available: http://www.iso.org/iso/catalogue_detail.htm?csnumber=59902

[49] K. Hauser, "The minimum constraint removal problem with three robotics applications," *Int. J. Robot. Res.*, vol. 33, no. 1, pp. 5–17, 2014.