Autonomous Robotics Manipulation for In-space Intra-Vehicle Activity

Shaun Azimi and **Emma Zemler**
Robotic Systems Technology Branch
NASA Johnson Space Center
2101 NASA Parkway
Houston, TX 77058
{shaun.a.azimi,emma.zemler}@nasa.gov

Robert A. Morris
Intelligent Systems Division
NASA Ames Research Center
Moffett Field, California 94035
robert.a.morris@nasa.gov

Abstract

Future space exploration missions will involve robotic assistants in habitable vehicles which autonomously adapt to complex, rapidly changing environments. Such systems will integrate deliberative behaviors such as task planning with more reactive behaviors like perception and control in order to maximize science return, and effectively perform logistics tasks, while complementing and enhancing the capabilities of human crews performing mission operators. This paper describes an autonomy technology demonstration project involving robotic arm manipulation tasks using an International Space Station (ISS)-inspired bio-medical science scenario. This domain offers potentially interesting problems in deliberative planning under uncertainty, and integrating task and motion planning. The demonstration system was implemented using the Robot Operating System (ROS) for inter-process communication between planning, perception and control, ROSPlan to implement deliberative behavior in the form of planning and execution, and an integration with a behavior modeling tool based on *Affordance Templates* in order to facilitate mapping high level actions with lower level behaviors.

Introduction

Future space exploration missions will require robotic assistants which adapt to complex, rapidly changing environments. Integrating deliberative behaviors such as planning to low level perception and control is crucial in order to maximize science return using in-space robotic systems, and to complement and enhance the capabilities of humans performing mission operators. NASA's Autonomous Systems and Operations (ASO) project develops and tests systems that assist crew members in performing science or logistic tasks in space.

This paper describes a technology demonstration of autonomous operation of a dexterous manipulator that is fixed in place. A mock-up of an International Space Station (ISS) science lab and an ISS-inspired bio-medical science scenario are used. The task planning problem requires action models for picking and placin objects, opening and closing storage devices, and place samples under a microscope to image. The system was implemented using the Robot Operating System (ROS) for inter-process communication between planning, perception and control (Quigley et al. 2009). ROSPlan (Cashmore et al. 2015) was used to implement deliberative behavior in the form of planning and execution. The objectives of the demonstration included integration of autonomous robotic operations with human activities in space, and to demostrate the potential to reduce human work load.

Following a summary of related work, we describe the ISS science scenario in detail, followed by a complete overview of the technical approach. We discuss the demonstration and lessons learned, and conclude with a summary of current activity.

Related Work

Research in integrating task planning with the dynamics of a robotic system for autonomous manipulation is currently being explored by a number of research teams. A detailed review of this research is beyond the scope of this paper; here we only mention a handful of the efforts that are to some extent related to our work.

There are a number of different approaches to solving the combinatorial challenges associated with combining two hard problems: task planning and motion planning. Some of them are based on the realization of the advantages afforded by leveraging the years of advances in heuristic search in classical planning. Thus, one set of approaches extends classical planning by introducing predicates into the plan model whose truth value is established by calling an external program that manages the motion or geometrical constraints on the overall problem (Srivastava et al. 2014). FFRob (Caelan Reed Garrett 2017) refines this approach by adapting heuristic search for solving classical planning problems to solve manipulation planning problems. Similarly, the approach in (Cohen, Chitta, and Likhachev 2010) builds a graph based on a small set of motion primitives to guide heuristic search of the joint angle state-space for manipulation problems. In (Ferrer-Mestres, Francès, and Geffner 2017), the task and motion planning constraints are compiled into classical AI planning problems, to avoid using expensive motion planners and collision checkers altogether, and where valid task plans end up as also valid robot plans.

For completeness, it should be noted that many approaches to combining task planning with lower-level functionality are not based on classical planning models. Many, such as (Bagnell 2012), are based on a behavior-based archi-

Figure 1: Astronaut Tracy Caldwell Dyson opening the MERLIN freezer in an express rack on the Space Station.

tecture, wherein low-level behaviors are composed into sequences of higher-level behaviors through a tree-like structure.

Finally, Astrobee (Bualat et al. 2018) is a free-flying robot that operates in the interior of the International Space Station (ISS), which is used to provide flight and payload controllers with a mobile camera/sensor platform to operations support.

In our view, autonomous systems that require a combination of deliberative and reactive capabilities will be based on a hybrid approach that combines high level task planning with behavior-based techniques. The approach used here, in particular, combines PDDL-based plan modeling with a mechanism for structuring behaviors based on the idea of *affordance templates* (Hart, Dinh, and Hambuchen 2015).

Science Scenario

Our demonstration scenario represents a generic scientific laboratory experiment that was derived from several recent experiments performed by astronauts on the International Space Station. Many biological and materials experiments utilize samples that are kept frozen until the experiment is conducted (see Figure 1). Most experiments involve data products that include microscopic or other forms of imagery, and many of them also specify some other processing, such as mixing, separation (such as by centrifuge), heating etc. We decided upon a generic procedure in which samples start out frozen in the freezer.

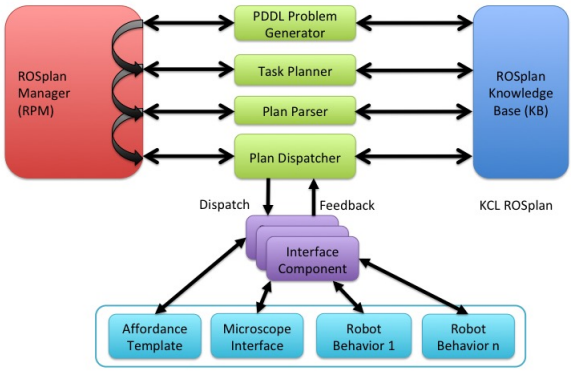

Figure 2: Integration of Task Planning with Robot Behaviors using ROSPlan

The requirements for accomplishing the scenario were:

- Remove samples (centrifuge tubes) from freezer
- Defrost all samples
- Run some of the samples through the centrifuge, but not all of them (a number set by user); and
- Image all of the samples in the microscope (some after being spun, some not).

The combined tasks comprised a complex enough scenario to demonstrate the need for integrated task and motion planning, as well as integration with other low-level robot behaviors.

The scenario represents a balance between three factors: (1) demonstration of capabilities useful for spacecraft Intra Vehicular Activity (IVA) science utilization applications; (2) constraints on the capabilities of the robotic system used; and (3) demonstrating the advantages of classical task planning over traditional scripted task sequencing.

It was decided that this scenario would be demonstrated on a real robot running a mockup of an ISS express rack, simulating an on-orbit operations. In the next section, we describe the hardware and software used to build the demonstration.

Technical Approach

Figure 2 provides a visual overview of the manipulation system. The system is arranged in a standard layered structure, with deliberative capabilities controlling and receiving feedback from lower, more reactive layers. The deliberative layers here consist of a ROSPlan manager (RPM), that commands and monitors the overall system, and the planning system for generating and executing task plans. The plan dispatcher commands and receives feedback from the robot behavior components, which are themselves composed of sequences of simple actions. The world state is routinely updated and stored in a database.

The system components will now be described in more detail.

```
; Pick a test tube from a stowage location.
    (:durative-action pick_stowage
        :parameters (?test_tube - test_tube ?
location
                            - stowage_location)
        :duration (= ?duration 15)
        :condition (and
            (at start (sawyer_available))
            (at start
(stowage_location_occupied ?location
                            ?test_tube))
        )
        :effect (and
            (at start (not (sawyer_available)))
            (at end
(stowage_location_available ?location))
            (at end (not
(stowage_location_occupied
                    ?location ?test_tube)))
            (at end (sawyer_holding ?test_tube))
        )
    )
```

```
; Place a test tube in a stowage location.
    (:durative-action place_stowage
        :parameters (?test_tube -
test_tube ?location
                                -
stowage_location)
        :duration (= ?duration 15)
        :condition (and
            (at start (sawyer_holding ?
test_tube))
            (at start
(stowage_location_available ?location))
        )
        :effect (and
            (at start (not
(stowage_location_available
                                    ?location)))
            (at start (not (sawyer_holding ?
test_tube)))
            (at end
(stowage_location_occupied
                                    ?location ?
test_tube))
            (at end (sawyer_available))
        )
    )
```

Figure 3: Pick and Place in PDDL

Task and Motion Planning

We used the ROS module ROSPlan (Cashmore et al. 2015) to implement deliberative behaviors and provide the interface with lower level control. ROSPlan contains a Knowledge Base that manages the evolving state of the world and interprets a model for planning. Planning problems are formulated by defining a PDDL 2.1 domain and populating the Knowledge Base with predicates that describe the initial state of the world. An example of PDDL pick and place action definitions for the manipulation domain is found in Figure 3. ROSPlan solves the resulting planning problem using heuristic forward search planners such as POPF (Coles et al. 2010) and LPG (Gerevini and Serina 2002). An example fragment of a plan for the manipulation scenario is found in Figure 4. In this snippet, the freezer door is open and the four test tubes are removed and placed into a stowage location to be thawed for a designated length of time. After that time, one test tube is placed into a centrifuge to be spun, prior to imaging, while another is placed under the microscope without spinning, and an image taken. A simple ROSPlan dispatcher executes each enabled action in the plan, i.e., actions whose preconditions are true.

Integration with Affordance Templates

Dispatching a fundamental manipulation action such as 'pick' or 'place' requires converting the action into a sequence of motions, i.e., solving a motion planning problem. To facilitate the integration of task and motion planning, the *Affordance Template* framework was used (Hart, Dinh, and Hambuchen 2015). An affordance template is a graphical representation that exists in a 3D immersive environment (such as RViz) to provide robot task goals (including spatial end-effector way- points, contact points for interaction, force magnitudes) and parameters (object scales, locations) in object-centric coordinate frames. If a template is placed in such an environment (a location that is said to afford the task behavior) with the associated goals and parameters, they can be sent to the robot's control system to perform the task.

```
0.0003:   (open_freezer) [47.0000]
47.0005:   (pick_freezer tube_a freezer_slot1) [15.0000]
62.0008:   (place_stowage tube_a stowage_location4) [15.0000]
62.0075:   (defrost_tube tube_a) [180.0000]
77.0010:   (pick_freezer tube_c freezer_slot5) [15.0000]
92.0013:   (place_stowage tube_c stowage_location3) [15.0000]
92.0072:   (defrost_tube tube_c) [180.0000]
107.0015:   (pick_freezer tube_b freezer_slot2) [15.0000]
122.0017:   (defrost_tube tube_b) [180.0000]
122.0020:   (place_stowage tube_b stowage_location2) [15.0000]
137.0022:   (pick_freezer tube_d freezer_slot6) [15.0000]
152.0025:   (defrost_tube tube_d) [180.0000]
152.0027:   (place_stowage tube_d stowage_location5) [15.0000]
167.0030:   (close_freezer) [45.0000]
212.0033:   (open_centrifuge_lid) [28.0000]
240.0035:   (pick_stowage tube_b stowage_location2) [15.0000]
255.0038:   (place_microscope tube_b) [15.0000]
270.0042:   (pick_stowage tube_d stowage_location5) [15.0000]
302.0040:   (image_no_spin tube_b) [180.0000]
332.0045:   (place_centrifuge tube_d centrifuge_slot1) [11.0000]
343.0048:   (close_centrifuge_lid) [22.0000]
365.0050:   (spin_centrifuge_single centrifuge_slot1 tube_d)
[180.0000]
482.0052:   (pick_microscope tube_b) [20.0000]
```

Figure 4: Plan Fragment for the Lab Science Scenario

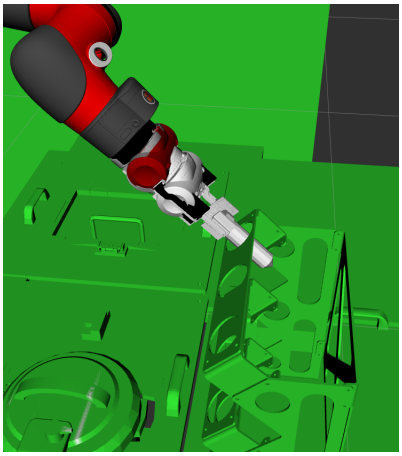

Figure 5: RVIZ visualization of robot at the first waypoint of the pick of a test tube.

To integrate task planning with the Sawyer robot, we decomposed the PDDL actions into easily achievable robotic behaviors, given the software capabilities of the robot. For motion planning and execution, a MoveIt! configuration for the robot and a MoveIt! compatible joint trajectory controller was used. However, using MoveIt! is still fairly low level. For example, it provides a mechanism to produce a valid motion plan from the current state of the robot to achieve a desired Cartesian pose for the end effector frame, while avoiding collisions against any known world collision objects. Ultimately, this level of functionality was able to be encapsulated into our behaviors by using the Affordance Templates, to build up larger behaviors such as picking up a test tube or opening the freezer.

Affordance Templates provide a way of describing an action that an object affords to the robot, such as picking up a test tube. The construct of an affordance template allows a human operator to define ordered end effector waypoints relative to the object to achieve a trajectory, such as pick for object test tube. Affordance Templates are implemented using an RVIZ plugin, where the object and waypoints have interactive markers, allowing the operator to perfect an approach to an object, and save the trajectory (see Figure 5). The waypoints consist of the Cartesian pose for the end effector and the state of the end effector, such as gripper open or gripper close. The Affordance Template framework integrates with our Sawyer planning server, allowing MoveIt! to generate robot trajectories to move the robot from its current state to a desired Affordance Template trajectory waypoint. At the behavior layer, we would plan to waypoints in the Affordance Templates trajectory and execute the manipulator trajectory motion, followed by commanding the gripper to the desired gripper state as part of the waypoint.

Our use of Affordance Templates allowed the robot operators to create a template for a test tube, with two different trajectories, one for pick and one for place. The operators were able to practice and adjust the waypoints until the trajectory could be used to interact with the object with reliability. Provided we know the position of the test tube to be picked, we subsequently have the ability to pick up any test tube by use of the template. Similarly, if we know the desired position of the test tube for stowage, we can invoke an affordance to place the test tube.

Finally, MoveIt! can only calculate the trajectory provided we maintain a good collision model of our world. As such, the apparatuses and work table were as added as collision objects to MoveIt. To allow the robot to consider a test tube or centrifuge lid presently in the gripper, we needed to attach the collision object to the end effector. By maintaining an up to date collision model of our world, we were able allow the robot to plan from its current state, without any concerns of unwanted interaction between the robot and other objects.

To integrate behaviors with ROSPlan, ROSPlan components were designed with a one to one relationship to the actions in the PDDL domain model. WHen ROSPlan dispatched an action to the desired component, the component would invoke the behavior required for completing the action. Upon successful completion of the behavior, we would update the ROSPlan Knowledge Base with the effects of the action, using sensor data as verification prior to updating as applicable.

Demonstration Platform Design

Our demonstration platform was inspired by existing scientific facilities aboard the International Space Station (ISS), but designed to be more generic for our initial research and development goals. Since our use case scenario specified that a robot be attached to an ISS EXPRESS rack and interact with rack payloads, we started out with standard rack and payload dimensions for our apparatus sizing. Since an ISS EXPRESS rack is effectively just a cabinet frame with rack mount rails Figure 6), we opted to build an equivalent sized frame using aluminum T-slot rails rather than acquire an expensive duplicate rack.

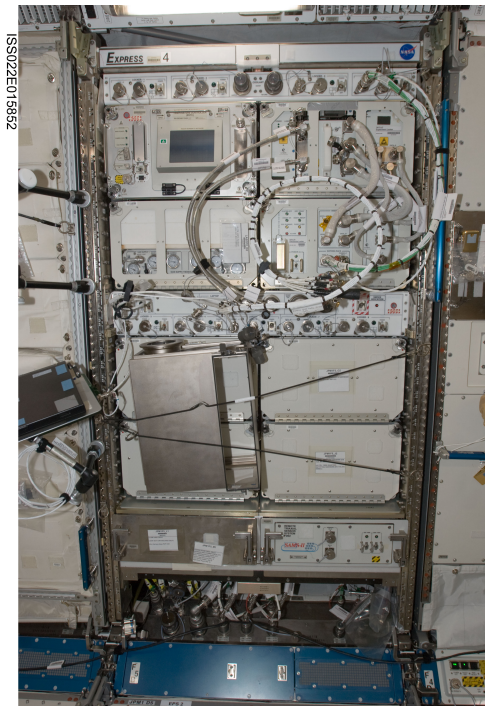

Figure 6: ISS Express Rack

Due to the reach limitations of the Sawyer robot, we made our T-slot frame half of the height of a real ISS rack (the other half would just take up more space since the robot could not reach it). We then designed three standard size rack-mount payloads to fit within our frame: (1) A mechanical mock-up freezer modelled after the MERLIN freezer used on ISS, (2) a mock up centrifuge with a twist-lock lid that is simulated by a modified kitchen appliance, and (3) a combination test tube rack and (functional) digital microscope Figure 7. All three payloads were designed with compliant interfaces that hold standardized containers, in this case 1.0inch diameter 50mL centrifuge tubes with caps (see Figure 8). Using standardized containers simplifies the design and control of the robot end effector, which has a lim-

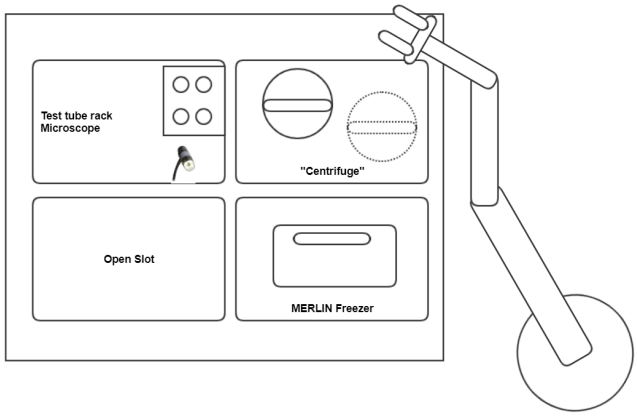

Figure 7: Cartoon Drawing of ISS Express Rack Design

Figure 8: Centrifuge Tube Similar to Those Used in the Demonstration

ited range of motion that can only accommodate a relatively small range of object sizes.

For apparatus components such as the freezer that were originally designed exclusively for human interaction, our mock-up version required some design accommodations for use with a robot. For example, the real MERLIN freezer has a small latch at the top of the door which is operated by flipping a small tab out of the way and squeezing two small levers together to unlatch and pull to open. We created an enlarged version of the latch and added a large handle that allows our robot to more reliably open and close the freezer door.

The test setup utilizes the Sawyer robot with electric parallel gripper manufactured by Rethink Robotics. We considered several factors in our choice of robot, such as native support for control using ROS and MoveIt trajectories, the availability of a manufacturer-supported Gazebo simulation environment, wide adoption by the research community, sufficient reach and payload capability, and cost. Based upon these factors, the leading contenders were Sawyer and the Universal Robotics UR5. One significant factor in favor of Sawyer is that it has seven degrees of freedom (DoF), compared with six for the UR5. This allows the robot to better handle working within more constrained environments, and matches more closely the configuration of a space-rated robot that is currently being developed for use aboard ISS (which will be the robot that we ultimately use for our application).

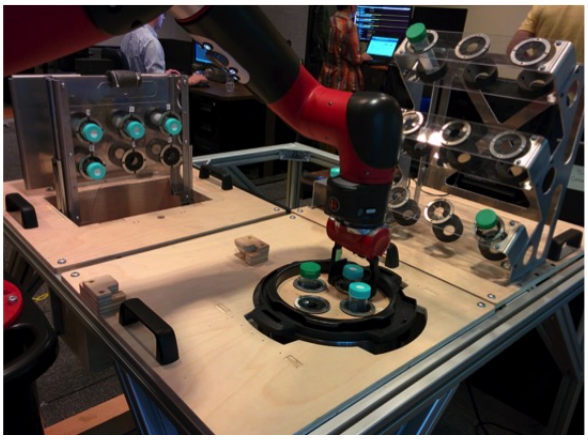

Figure 9: Robotic Manipulation Platform With Sawyer Robot

Discussion and Lessons Learned

The scenario described above was successfully demonstrated on the robotic platform using the Sawyer arm, using ROSPlan for task planning and execution. With proper tuning of input parameters, optimal plans (based on makespan) could be generated by LPG in a matter of a few seconds, and became the planner of choice for us (over POPF; since our goal here was not to compare PDDL planners, we offer no performance statistics here). Plan lengths were in the range of a few dozen, and the plan model contained roughly 10 durative actions and 50 predicates. Plans were optimized by plan length and representing duration allowed for parallelization.

A simple user Interface was developed to manually set plan goals as well as to monitor execution. In particular, we simulated an microscopic imaging event that results in a faulty (e.g. blurred) image. In this case, the user could manually command a re-take of the image.

The following personal observations summarize the challenges involved in deploying AI planning and execution for solving potentially hard manipulation problems.

First, we found that planner performance was sensitive to small changes in model or problem definition changes. For example, simply increasing the problem from using four test tubes to five test tubes sometimes resulted in planner time out or failing to find a solution. To address planner performance we sometimes resorted to adding more constraints into the plan model to assist in search guidance, but this effort was based too often on trial-and-error, which slowed development.

Similarly, there were challenges in modeling to avoid redundant actions (like opening and closing the freezer door more that once before removing the tubes). Again, such a problem could be solved either through plan model redesign (adding constraints to force the solver to solutions without redundant actions) or through optimization (favoring plans with minimal number of actions). Again, finding the best tradeoff between search and model structure was a challenging undertaking.

Another challenge was effective translation between discrete and continuous states in order to properly carry out an action cancellation. For example, if a cancel command were to be issued while opening the centrifuge, there are two issues: the robot is holding the lid in its gripper, and the lid is in between the closed and open states. It is essential that the world be returned to a state that corresponds to a ROSPlan knowledge base state variable assignment. This is especially important if replanning is to be performed. For re-planning, it would be best for the robot to be free and for the centrifuge lid to be either fully open or fully closed; therefore the ROSplan action component definition should ensure that one of those states is achieved if possible and update the knowledge base accordingly (as open, closed, or unknown).

Finally, it is often useful to group planner-level goals with different levels of abstraction. For example, set up the experiment or unload the container goals may consist of a set of sub-goals than must be solved sequentially. Although PDDL offers some level of support for abstraction of this kind, there was a sense that plan languages that support building more domain-specific, hierarchical plan models (e.g. along the lines of (Kaelbling and Lozano-Perez 2011)) would have been beneficial.

Summary and Future Work

This paper has described a successful technology demonstration of integrated task planning for a in-space intravehicle activity application of a robot manipulator. The approach combines task planning and execution using PDDL planning and ROSPlan, with motion planning and control based on an Affordance Template model of robot behavior.

Building an effective manipulator that combines deliberative and reactive behaviors requires a careful blend of plan modeling techniques; automated planner and plan representation (e.g. sequential, hierarchical, temporal); plan dispatching and execution; and goal reasoning and high level mission management. It also requires a framework for integrating deliberative models and decision-makers with lower level models and behaviors. The primary focus for this paper has been on the integration side, ensuring an effective mapping of plan actions with behaviors and skills. To accomplish this, we tended to simplify the problem by assuming (mostly) a deterministic world with no exogenous events and uncertainty. Current plans for this project include expanding the use case scenario, robot and apparatus described in this paper to demonstrate a combination of enhanced deliberative capabilities, with tighter integration with lower-level behaviors.

Under discussion is a scenario in which two or more high level tasks are undertaken in parallel, with priorities assigned to each. A 'goal manager' would monitor the accomplishment of goals to ensure they are executed in a order corresponding to their priority. We are also planning to demonstrate the system's ability to be robust to different types of unexpected exogenous events, or unexpected effects of actions. A simple example would be to respond to emergencies such as fire alarms, which would require preemption of all current planned activities and executing a contingency response plan. Another focus is to have a better mapping between actions in the configuration space of the robot with actions in the plan space, in order to effectively handle 'mid-action' failures in the plan action (as discussed in the previous section). Finally, we are exploring ways to enable 'pro-active' behaviors through incorporating models that anticipate future plan failures (such as those resulting from violating time constraints on plan execution).

Acknowlegements

This work was performed under NASA's Autonomous Systems and Operations (ASO) Project. Thanks to Jeremy Frank (ASO Project Manager), as well as the rest of the ASO team: Mark Paterson, Eric Gallagher, William Baker, Michael Park, and Kimberly Hambuchen.

References

Bagnell, J. a. 2012. An Integrated System for Autonomous Robotics. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2955–2962.

Bualat, M. G.; Smith, T.; Smith, E. E.; Fong, T.; Wheeler, D.; and the Astrobee Team. 2018. Astrobee: A new tool for iss operations. In *Proc. SpaceOps (AIAA 2018-2517)*.

Caelan Reed Garrett, Tomas Lozano-Perez, L. P. K. 2017. Ffrob: Leveraging symbolic planning for efficient task and motion planning. *The International Journal of Robotics Research*.

Cashmore, M.; Fox, M.; Long, D.; Magazzeni, D.; Ridder, B.; Carrera, A.; Palomeras, N.; Huros, N.; and Carreras, M. 2015. Rosplan: Planning in the robot operating system. In *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling*, 333–341. ICAPS.

Cohen, B. J.; Chitta, S.; and Likhachev, M. 2010. Search-based planning for manipulation with motion primitives. *2010 IEEE International Conference on Robotics and Automation* 2902–2908.

Coles, A. J.; Coles, A.; Fox, M.; and Long, D. 2010. Forward-chaining partial-order planning. In *Proceedings of the 20th International Conference on Automated Planning and Scheduling, ICAPS 2010, Toronto, Ontario, Canada, May 12-16, 2010*, 42–49.

Ferrer-Mestres, J.; Francès, G.; and Geffner, H. 2017. Combined task and motion planning as classical ai planning. *CoRR* abs/1706.06927.

Gerevini, A., and Serina, I. 2002. Lpg: A planner based on local search for planning graphs with action costs. In *Proceedings of the Sixth International Conference on Artificial Intelligence Planning Systems*, AIPS'02, 13–22. AAAI Press.

Hart, S.; Dinh, P.; and Hambuchen, K. A. 2015. The affordance template ros package for robot task programming. *2015 IEEE International Conference on Robotics and Automation (ICRA)* 6227–6234.

Kaelbling, L. P., and Lozano-Perez, T. 2011. Hierarchical task and motion planning in the now. In *IEEE Conference on Robotics and Automation (ICRA)*. Finalist, Best Manipulation Paper Award.

Quigley, M.; Gerkey, B.; Conley, K.; Faust, J.; Foote, T.; Leibs, J.; Berger, E.; Wheeler, R.; and Ng, A. 2009. Ros : an open-source robot operating system. In *IEEE International Conference on Robotics and Automation (ICRA 2009)*.

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