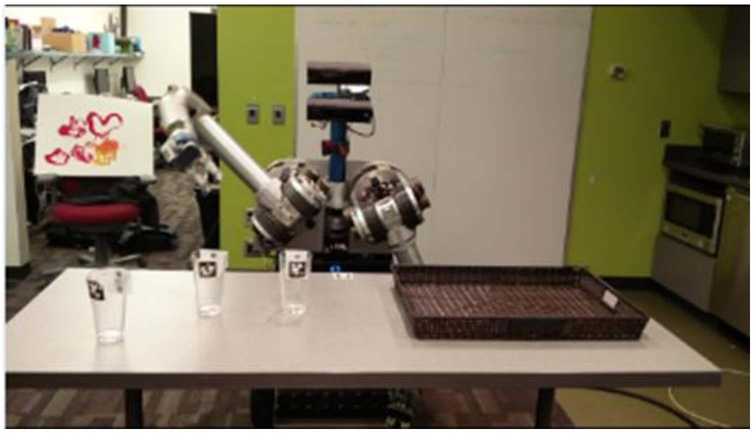
COMBINING TASK AND MOTION PLANNING FOR COMPLEX MANIPULATION

16-662: ROBOT AUTONOMY PROJECT



Team:

Gauri Gandhi
Keerthana Manivannan
Lekha Mohan
Rohit Dashrathi
Richa Varma

ABSTRACT

Robots performing household tasks is a big challenge, where the operator/user needs to give sequential commands to the robot for each subtask. In order to perform a complicated task in say, a kitchen environment, task planners are needed that can reason over very large sets of states by manipulating partial descriptions, while geometric planners operate on completely detailed geometric specifications of world states. Home Exploring Robot Butler (HERB) currently uses a sequential task planner for table clearing which plans for the entire task and obtains a feasible global plan before beginning execution. We aim to implement a task planner that reduces the planning time by breaking the high level task into subgoals, making choices and committing to them, greatly reducing the length of plans. This approach is suitable for complex tasks where strict optimality is not crucial. In this report, we present the key features of HPN and its advantages that led us to use these key features (fluents, operators, etc.) in a sub-version that uses a breadth-first approach to plan in the now.

INTRODUCTION

Project Description

Consider a robot unloading the dishwasher. Currently a user may have to specify a series of commands to make the robot carry out this task: drive to the dishwasher, open the dishwasher, pull out the top rack, pick up the cup, open the cupboard, put the cup into the cupboard, etc. We aim to enable HERB to take a high level task input, i.e. unload the dishwasher, and build a hierarchical plan to select, sequence and execute subtasks in the now. This project aims to use the ideas of HPN [1] to define a table clearing task and carry out interleaved planning and execution. The goal is to enable HERB to clear a table by picking up three cups placed in random configurations and placing them on a tray kept on the table.

Motivation

Robots in home environments need to deal with real time constraints. The main challenge we face here is the complex integration between a high-level task planner that selects the ordering of subtasks with low-level motion planners that generate arm motions to execute each subtask. Moreover, if the motion planning fails, for example, when the robot picks up the first two cups successfully but there is no motion plan to execute the subsequent tasks. Thus, there is a need for a robust task planner for a complicated sequence of tasks.

RELATED WORK: HIERARCHICAL PLANNING IN THE NOW: (HPN)

Hierarchical Planning in the Now is an approach to reduce planning time and plan length for carrying out a complex manipulation task. It operates on detailed, continuous geometric representations and does not require a-priori discretization of the state or action space. It is aggressively hierarchical. It makes choices and commits to them, exponentially decreasing the amount of search required.

HPN can work effectively with non-determinism in the environment or in the low-level controllers where planning in detail far into the future will mostly be wasted, due to the inability to predict exactly what will happen. Planning ‘in the now’ will construct a plan at an abstract level, commit to it, and then recursively plan and execute actions to achieve the first step in the abstract plan without constructing the rest of the plan in detail.

One of the risks associated with this approach is that the abstract plan might not be executable: the way that the first step is carried out could make it impossible to carry out subsequent steps, without undoing the results of earlier steps. This scenario is avoided by constraining the abstract plan to be serializable, i.e. for any step, there exist realizations for next steps.

Interleaved planning and execution is a relatively standard regression-planning algorithm, based on an A* search that works backward from the goal, generating sub-goals that are the weakest precondition of the goal under each applicable action. It is used to solve single planning sub-problems. The architecture can be thought of as doing a depth-first traversal of a planning tree, and is implemented as a recursive algorithm.

Representation

- Fluents

The logical aspects of a domain are characterized using fluents. A fluent is a symbolic predicate applied to a list of arguments, which may be variables or constants.

- World States

A world state is a completely detailed description of both the geometric and non-geometric aspects of a situation. The only requirement is that, for each fluent type, there is a procedure that will take a list of ground arguments and return the value of that fluent in the world state.

- Goals

A goal for our planning and execution system is a set of world states, described using a conjunction of ground fluents. During the course of regression-based planning intermediate goals are represented as conjunctions of fluents as well.

- Operators

A planning domain is characterized by a set of primitive actions. Each of these primitive actions is characterized by one or more operator descriptions. Each operator description can be used at multiple levels of abstraction, from the most abstract description to the most concrete one.

In a discrete domain, we can define the operators in a STRIPS-style form:

$$\begin{array}{l} F(A_1, \ldots, A_n) = V\text{:} \\ \quad \textbf{exists: } B_1, \ldots, B_k \\ \quad \textbf{pre: } \phi_1, \ldots, \phi_m \\ \quad \textbf{sideEffects: } \psi_1, \ldots, \psi_l \\ \quad \textbf{prim: } \pi \\ \quad \textbf{cost: } c \end{array}$$

where F(A1; : : : ;An) = V is the target fluent, the Ai and V are variables or constants, the Bi are variables and c is a positive real cost.

OUR APPROACH

Step 1: Literature review

Our first step towards implementing a new task planner on HERB was to study literature on three algorithms: HPN[1], SAHTN[2] and HBF[3]. We found HPN to be an interesting approach to the problem which is already implemented for the PR2 robot to carry out various complex tasks in a kitchen environment. With the advantages mentioned in the preceding sections, we chose HPN as a suitable solution to our problem.

Step 2: Defining the Table Clearing task: Representation

FLUENTS

- In(O,R): Has value True if object O is entirely contained in region R, otherwise False
- Holding(): Has value None if the robot is not grasping an object; otherwise the object being grasped
- ClearX(O[],R): Has value True if region R is not overlapped by any object except those in the list Os, otherwise False
- Mindist(O)/Sumdist(O): A metric fluent that determines if the object O should get a clear flag to be picked or not

WORLD STATES

In our implementation, the world state is represented by a configuration of the robot, a fixed table and a set of objects on the table, each of which has attributes including pose, shape, whether it is grasped by the robot, etc.

GOALS

The goal of having object A, B and C to be clean and in the washer can be articulated as:

In(A; tray) = True ^ In(B; tray) = True ^ In(C; tray) = True

This is the highest level goal which is broken down into sub-goals to be evaluated and executed by the task planner iteratively.

OPERATORS

- Pick()
- Place()

These are the two primitive actions in our task definition that the low level planners in the robot have the capability to plan and execute.

Step 3: Planner implementation

We implemented a sub-version of the HPN planner which is formulated on the basis of key concepts of HPN such as fluents, operators, world states and goals described in the previous step. Our planner does not proceed on the basis of abstraction levels as defined in the HPN method. In traditional HPN, operator descriptions include an abstraction level: and the action is executed only when the operator is at its most concrete level of abstraction. This is the idea behind postponing pre-conditions for sub-goals in order to form a hierarchical plan execution tree. Since implementing the entire HPN architecture was beyond the scope for this project in terms of schedule and amount of work involved, we implemented a breadth first approach that uses precondition fulfillment criteria to postpone sub-goals in the planning domain.

Following are additional details of our implementation:

1. Metrics used:

PICKING

- Minimum Distance from the End Effector
- Maximum Distance from the other objects on the table
- Minimum Distance from the goal region
- Distance between objects greater than the size of end effector
- Maximum clear area around the object

PLACING

- Random selection of goal points in goal region
- TSR Library ensures to return only collision free solutions in Goal region

2. Motion planner used:

TSR (Task Space Region) Library was used for low level motion planning in our task. The TSR gives a number of feasible configurations for the arm to carry out the pick and place operations on the 'glass' kinbody.

RESULTS

The following results were observed for different configurations of cups placed on the table. The ley performance measure recorded was the plan time. It was observed that our planner tends to trade off robustness for speed, but the task execution was successful 50% of the time.

Our version (sec)	HGPC (sec)
79.736	111.165
77.782	120.368
81.774	115.180
Average: 79.764	Average: 115.571

Table.1: Planning times for different configurations of objects

SIMULATION RESULTS

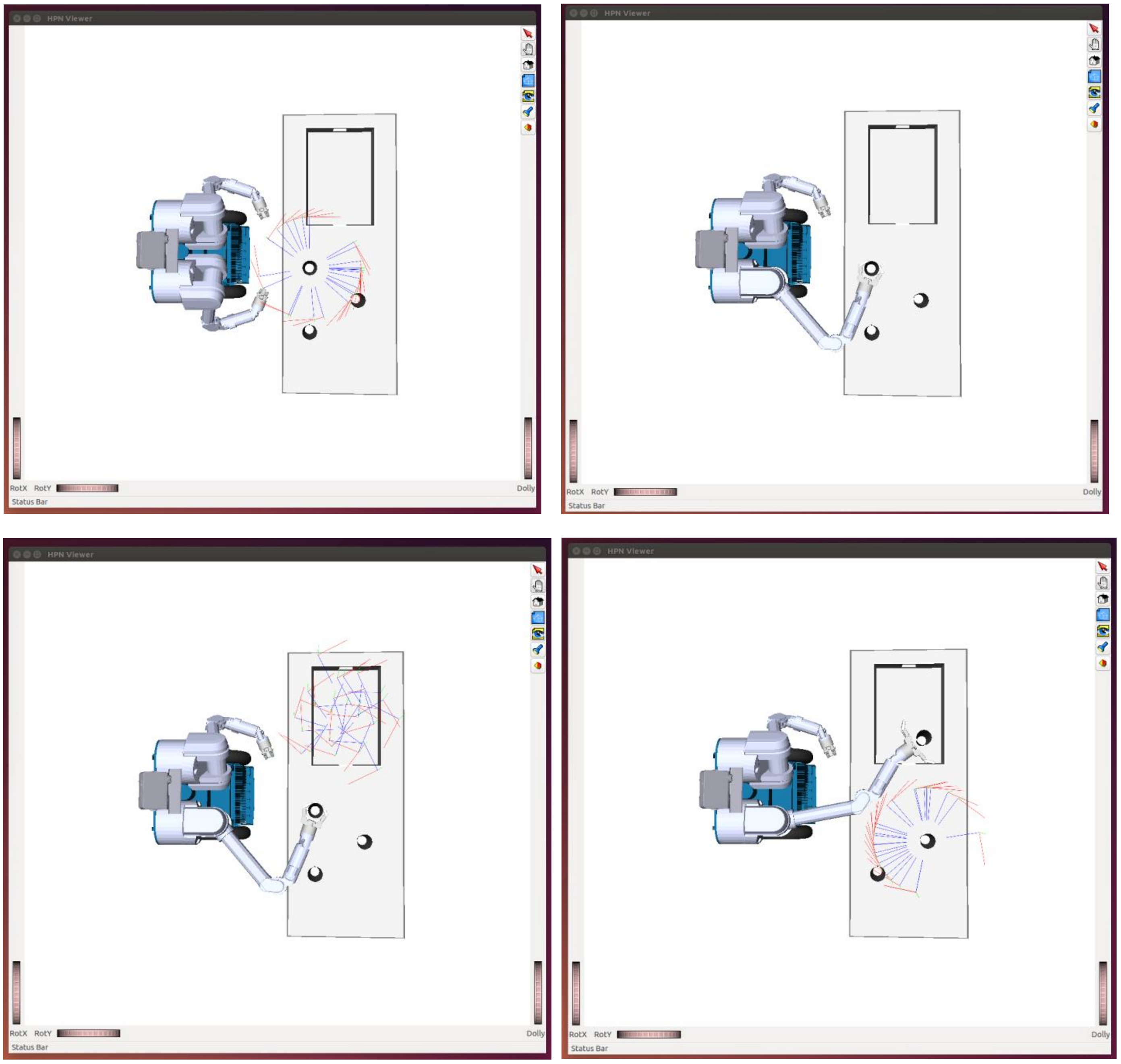
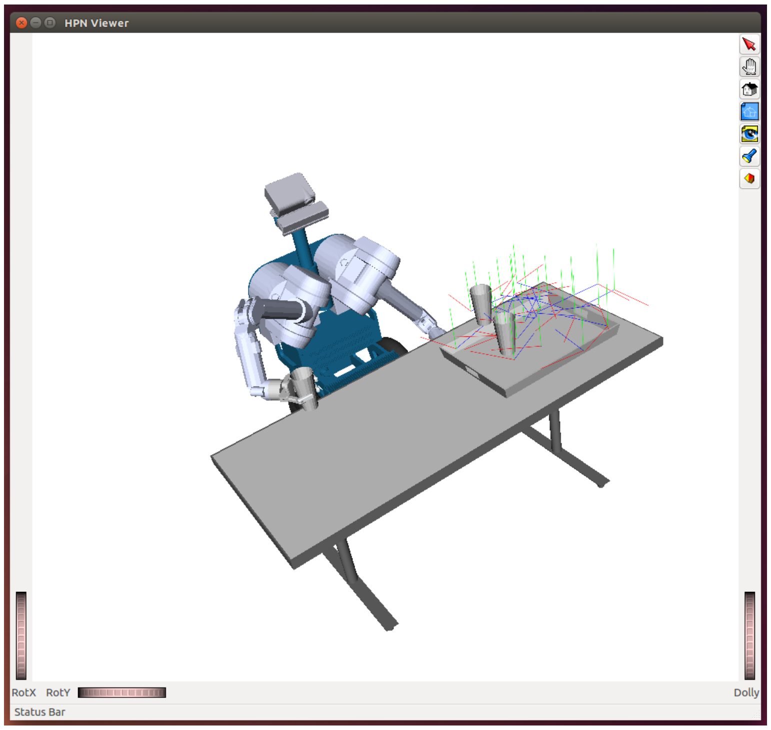


Fig.1 : HERB performing pick and place for the first cup (A)



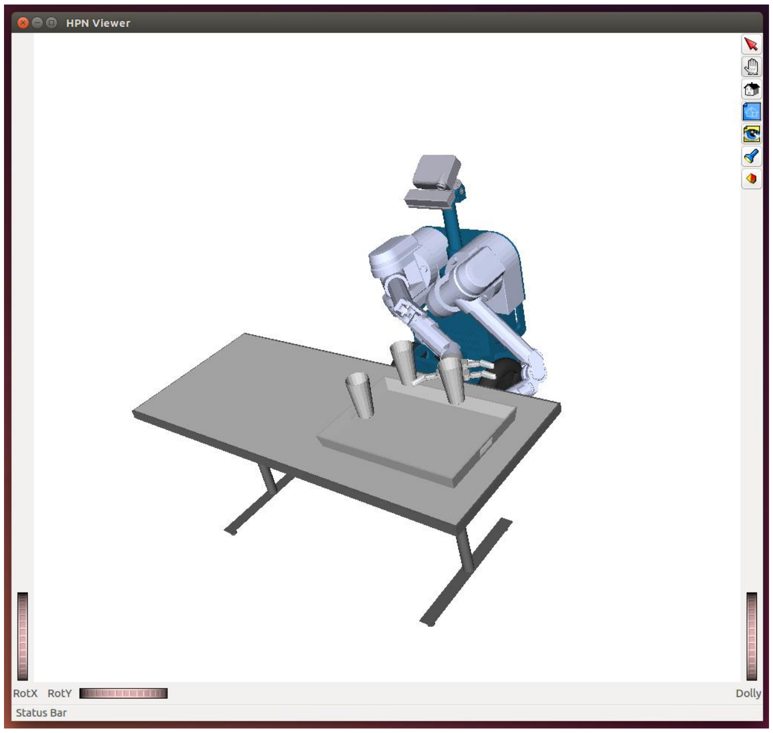


Fig.2: HERB placing final cup on the tray to complete the task successfully

IMPLEMENTATION ON HERB (REAL ENVIRONMENT)

As we switched from the simulated environment to the real robot, we faced a lot of issues that were hindering the testing of the task planner. To overcome those issues we tuned certain parameters in the code for troubleshooting. Mentioned below are few of the challenges we faced with the hardware and the strategies that we followed to overcome them:

1. Inaccuracies in HERB's perception system:
HERB uses an April Tag based perception system to determine the exact poses and orientation of the kinbodies in the environment. In most of the cases, the readings are inaccurate.
To solve this issue, we created two methods: place_on(to place the objects at 20cm above the table) and place_after(to place the objects 10 cms beyond their detected pose) in the code that allowed us to reduce the distance by a few centimeters between the actual position and the detected pose of the object.

2. Hand's preshape was hitting other objects:
Herb's hand's preshape was set to [0, 0, 0, 0] corresponding to the three fingers and the arm rotation. This was causing the other objects on the table to displace when the hand would open before picking or after placing.
Therefore, we changed the preshape to [0.9, 0.9, 0.9, 0] so that the hand takes minimum area while in operation.

3. Grasp operation was displacing the target object:
The grasp function from the TSR library is designed to grasp the object from the detected position. This was causing the hand to hit the target glass while picking and thus resulting in an unsuccessful grasp.
Therefore we decided to use the push grasp with a push distance of 5 cm.

4. Robot hitting other objects on the table while executing planned trajectory:
We observed that the trajectory planned by the robot to move from start location to the goal location would hit the other objects due to inaccurate perception.
Therefore we added a lift function from the TSR library with a lift distance of 30 cm so that all the trajectories are planned from a lifted start point.

FINAL VIDEO LINKS AND DESCRIPTIONS:

HERB performing the task using the Min_Dist metric (our HPN version) : Cup configuration 1

HERB performing the task using the Min_Dist metric (our HPN version) : Cup configuration 2

HERB performing the task (identical cup configuration) using existing HGPC planner on HERB

CONCLUSION AND FUTURE WORK

One of the challenges in robot planning lies in representing the world state and actions in complex domains, dealing with the computational complexity of planning and coping with the challenges of uncertainty. Symbolic task planning focuses on planning over a wide range of domains and has developed effective techniques for abstraction and factoring to deal with large domains; robot motion planning focuses on planning robot motions in complex geometric domains. This approach aims to bridge the gap between task and motion planning. There are two aspects to this approach:

- A tight integration between the logical and geometric aspects of planning. Given this representation and some simple mechanisms for geometric inference, we can characterize the pre-conditions and effects of robot actions in terms of these logical entities.

- By postponing the preconditions of actions to create an abstraction hierarchy that both limits the lengths of plans that need to be generated and limits the set of objects relevant to each plan.

These components trade off some generality for efficiency, but offer desirable results in terms of planning time and shorter plans. Both tightly integrated logical and geometric representations and hierarchical action decompositions are likely to play central roles in building autonomous robot systems.
This project was the first step towards implementing a hierarchical task planner on HERB for complex manipulation tasks. We were successful in demonstrating a reduction in plan time compared to the existing sequential planner.

Future work includes implementing operators with abstraction levels and using recursive planning to build the plan execute tree for the task. After this is done, costs and side effects of actions can be added to make the planner more efficient and robust.

REFERENCES

[1] Hierarchical Task and Motion Planning in the Now, *Leslie Pack Kaelbling and Tom´as Lozano-P´erez*
[2] Combined Task and Motion Planning for Mobile Manipulation, *Jason Wolfe, Bhaskara Marthi, Stuart Russell*
[3] Backward-Forward Search for Manipulation Planning, *Caelan Reed Garrett, Tomas Lozano-P ´ erez,, Leslie Pack Kaelbling*