Hierarchical Task and Motion Planning in the Now

Presenter: Zifan Xu

11/09/2021

Manipulation Tasks in the Real World

Motion planning is enough to solve tasks with single objective (e.g. grasp a mug)

Nvidia

Real world tasks need executions of sequence of objectives in a correct order and long time horizon

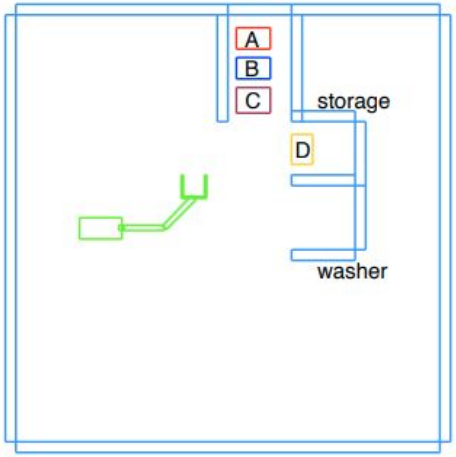
Google image

Combining Task and Motion Planning (TAMP)

- **Planning in task level: wash clothes A and put A in storage**

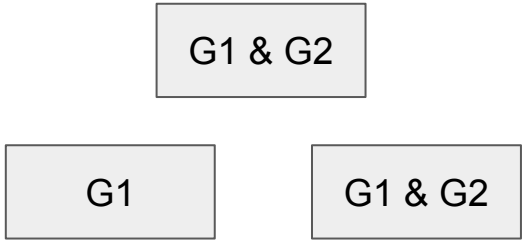
Pick(C) -> Place(parkingC) -> Pick(B) -> Place(parkingB) -> Pick(A) -> Place(washer) -> Wash -> Pick(D) -> Place(parkingD) -> Pick(A) -> Place(storage)

- **Motion planner translate tasks to actual motion plans**
- **Combining task planning and motion planning is not trivial!**
  - Non-determinism in the low-level motion planner
  - Task planning under continuous geometric states

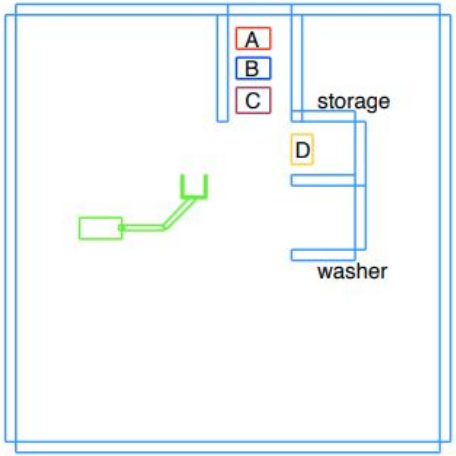


Hierarchical Task and Motion Planning in the Now

- **Non-determinism in the low-level motion planner**
  - Aggressive hierarchical planning: make choice and commit it
  - Constrain abstract plan steps to be serializable



- **Task planning under continuous geometric states**
  - Use “‘suggesters’” to propose appropriate discretized states



A Concrete Example

Goal: *In(a, storage) ∧ Clean(a)*

Plans: *Pick(C) ->*

Place(parkingC) -> Pick(B) ->

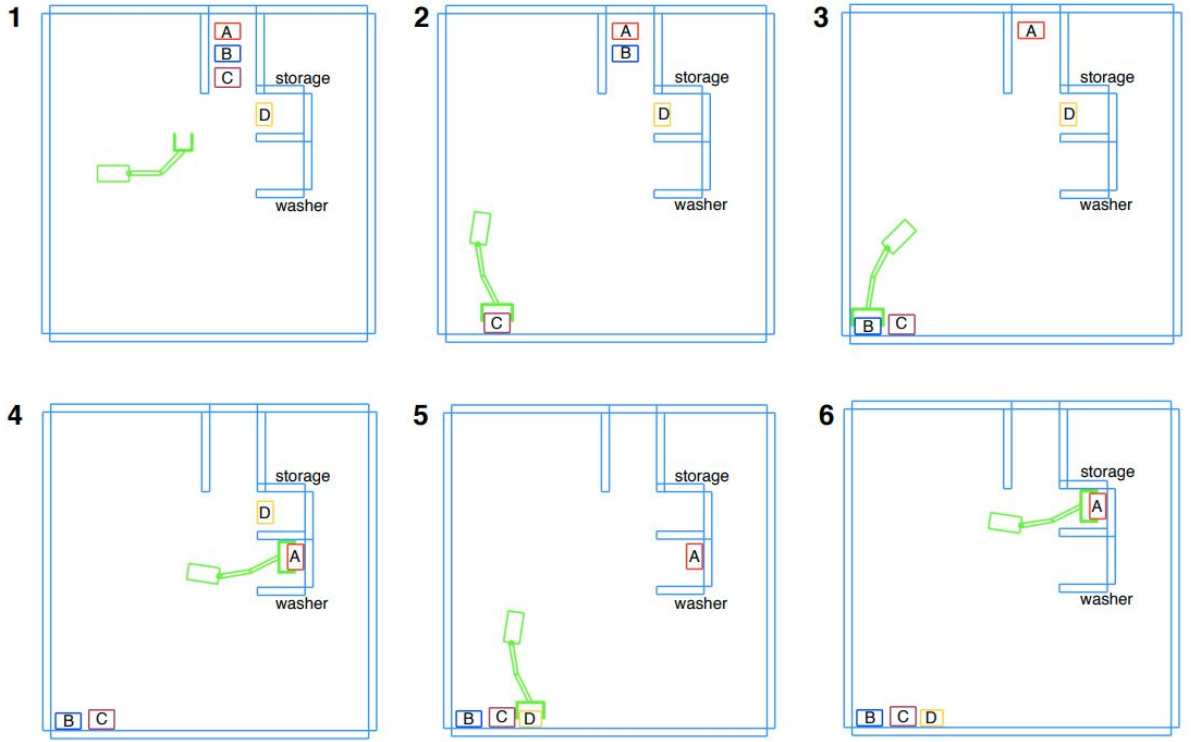
Place(parkingB) -> Pick(A) ->

Place(washer) -> Wash ->

Pick(D) -> Place(parkingD) ->

Pick(A) -> Place(storage)

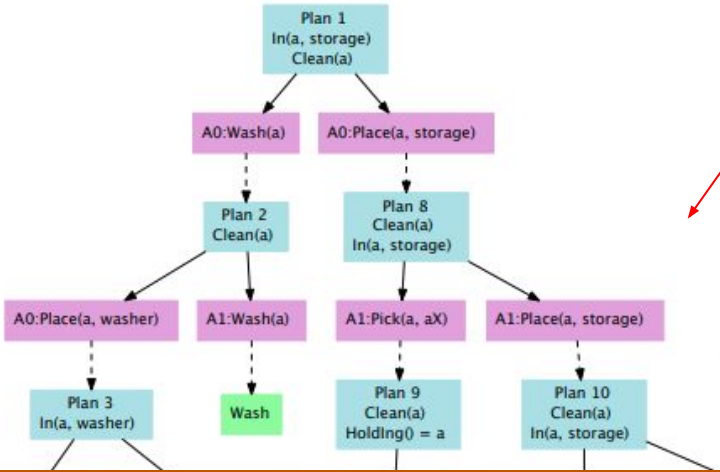
How does the robot come up with this plan?

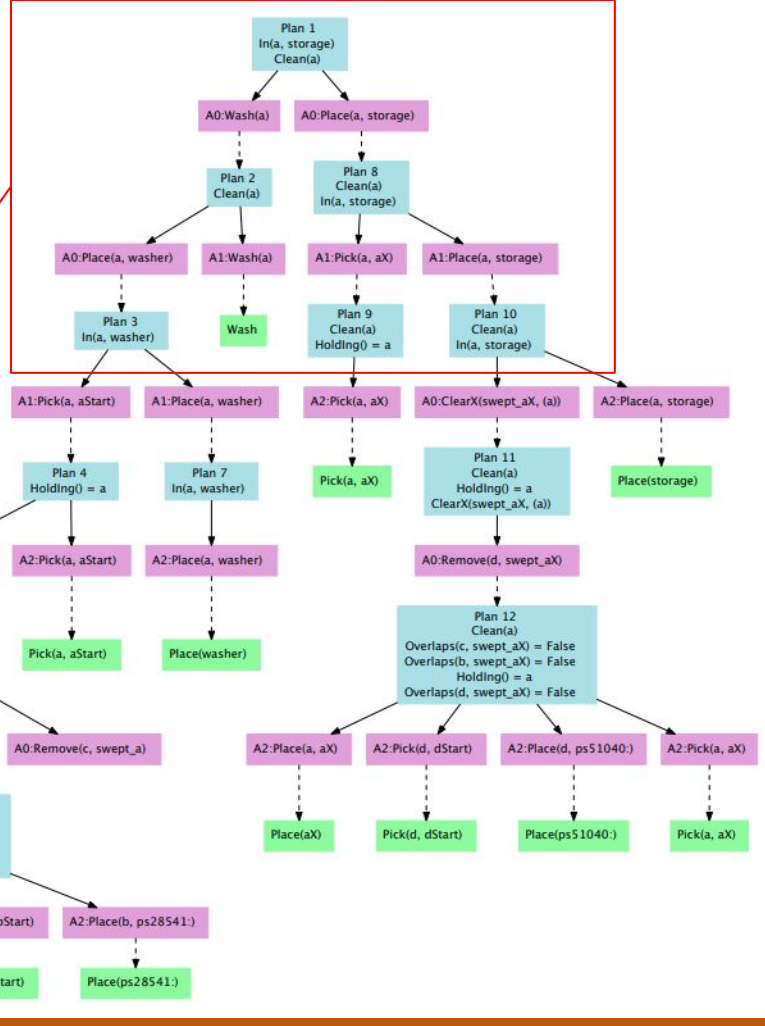


Hierarchical Planning and Execution Tree

Naive intuition:
Depth first search with an order of preconditions to decide which child node to descend first.

Until it reaches the leaf node of primitive action that will be solved by motion planning





Planning Domain Description

Fluents: symbolic predicate that characterizes the logics aspect of the domain

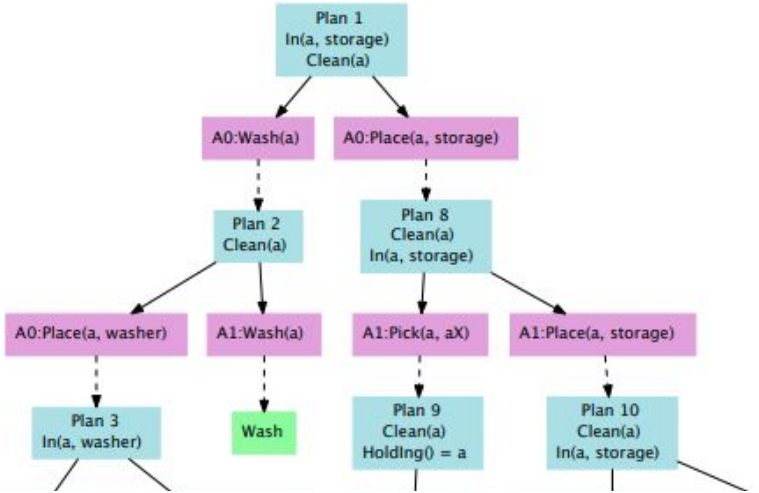
In(O, R), Overlaps(O, R), ClearX(R, Os), Holding(), Clean(O)

Goals: conjunction of fluents with values

In(a, storage) = True ∧ Clean(a) = True

Operations: primitive actions

Pick(O), Place(O, R), Wash()



Operator Description

STRIPS-style form:

$F(A_1, \ldots, A_n) = V$:
exists: $B_1, \ldots, B_k$
pre: $\phi_1, \ldots, \phi_m$
sideEffects: $\psi_1, \ldots, \psi_l$
prim: π
cost: c

Target fluent: $F(A_1, \ldots, A_n) = V$

Preconditions: $\phi_1, \ldots, \phi_m$ represented as fluents

Side effects: $\psi_1, \ldots, \psi_l$ represented as fluents

Primitive action: π

Semantics: if the primitive actior π is executed in any world state *s* in which all of the φ fluents hold, then the resulting world state will be the same as *s*, except that any fluent mentioned as the target fluent or a side effect will have the value specified by those fluents.

Beyond Standard Operator Descriptors

$Holding() = O$:

define: $Ts = \{T : ClearX(T, X) \in \textbf{goal} \wedge O \notin X\}$

exists: $L \in \{Location(O, \textbf{start}),$
$SuggestParking(O, Ts, \textbf{start})\}$
$P \in SuggestPaths(O, L, home, \textbf{start})$

pre: 0. $Holding() = nothing$
0. $In(O, L) = True$
2. $ClearX(sweptVol(P), [O]) = True$

sideEffects: $\forall L'.In(O, L') = False$

prim: $Pick(O)$

Abstraction level

Suggesters:
(a) not conflict with the current planning goal;
(b) ensure that the preconditions of the operation can be effectively serialized.

Other Operators

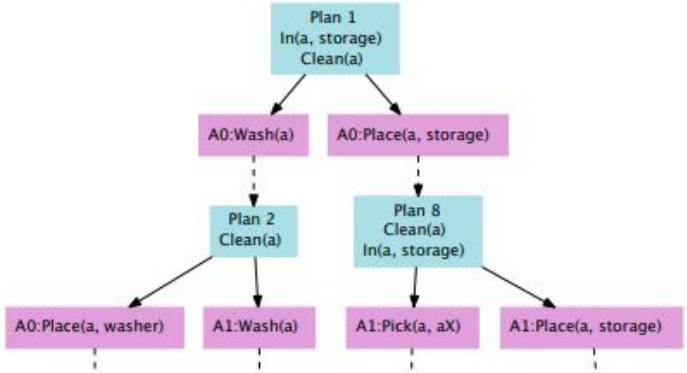
$In(O, R) = True$:
define: $Ts = \{T : ClearX(T, X) \in \textbf{goal} \wedge O \notin X\}$
exists: $P \in SuggestPaths(O, R, home, \textbf{start})$
pre: 1. $Holding() = O$
2. $ClearX(sweptVol(P), [O]) = True$
sideEffects: $Holding() = nothing$
prim: $Place(R)$

$Overlaps(O, R) = False$:
define: $Ts = \{T : ClearX(T, X) \in \textbf{goal} \wedge O \notin X\} \cup \{R\}$
exists: $L = SuggestParking(O, Ts, \textbf{start})$
pre: 1. $In(O, L) = True, ClearX(L, [O]) = True$
prim: none

$ClearX(R, Os) = True$:
pre: 1. $\forall X \in Objects - Os : \ Overlaps(X, R) = False$
prim: none

$Clean(O) = True$:
pre: 1. $In(O, \text{WASHER})$
prim: $Wash()$

Hierarchy



Preconditions with abstraction level:

postponing Preconditions with high abstraction levels

pre: $p_1, \ldots, p_n$
prim: o

→

pre: $p_1, \ldots, p_{n-1}$
prim: achieve p_n maintaining $p_1, \ldots, p_{n-1}$; o

The new operator description may not be true:

(a) Suboptimality: $p_1, \ldots, p_{n-1}$ may not be possible without p_n

(b) Achieving p_n may additional side-effect

$Clean(O) = True$:
pre: 1. $In(O, \text{WASHER})$
prim: $Wash()$

$Clean(O) = True$:
pre:
prim: $In(O, \text{WASHER})$ $Wash()$

$In(O, \text{WASHER})$

$Clean(O) = True$:
pre:
prim: $Wash()$

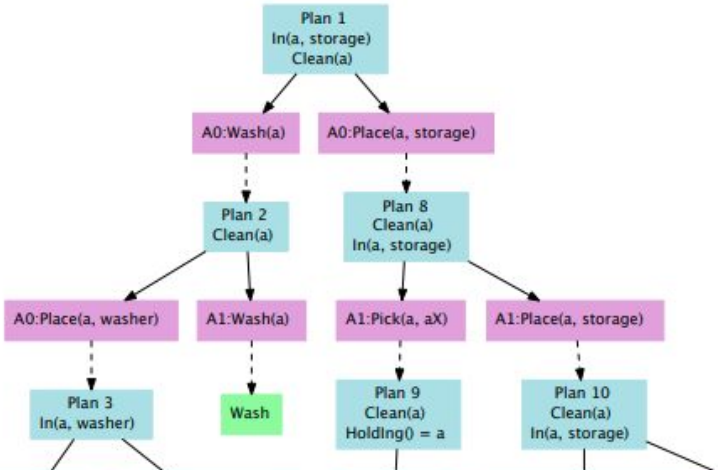
Algorithm

Calling HPN recursively:

```
HPN(currentState, goal, operators, absLevel, world):
    if holds(goal, currentState):
        return TRUE
    else p = PLAN(currentState, goal, operators, absLevel)
        for (o_i, g_i) in p
            if prim(o_i):
                currentState = world.execute(o_i)
            else HPN(currentState, g_i, operators,
                     NEXTLEVEL(absLevel, o_i), world)
```

Leaf node of a single primitive action

Hierarchy: return a list of operators and goals $((o_1, g_1), \ldots, (o_n, g_n))$ where the o_i are operator instances, $g_n = goal$, g_i is the weakest precondition of g_{i+1} under o_{i+1}



Correctness

Theorem:
If

(1) The planning domain description (PDD) specified by operators *ops* at the most concrete abstraction level H^* is a complete and correct formalization of the primitive actions of domain *w*;

(2) **start** has static connectivity in that domain;

(3) *G* is reachable from **start**,

Then, executing **HPN**(**start**, *G, ops,* H_0 , *w*) will cause world *w* to be in a state $s \in G$.

The theorem guarantees if a goal state was reachable from the starting state under some sequence of operations, that **HPN** will eventually cause the system to reach a goal state.

Partial ordering of abstract level mapping:
Abstract -> concrete
(H_0, …, H_i, …, H^*)

Static connectivity: a state *s* has static connectivity in a domain, if all states s_1 that are reachable from *s* are also reachable from any s_2 that is reachable from *s*

Empirical Results

Domains:

- Wash: variants of our concrete example
- Household: large (6 rooms), complex operations (’vacuum’ and ‘mop’)
- Swap: interchanging the locations of two blocks (non-serializable goals)

Achievements:

- Successfully plans with different challenges presented in these domains
- Plans with no or few redundant steps

Domain	Num	Longest	Steps
swap	22	4	8
wash	14	4	13
wash all	26	6	22
clean house	89	4	36
clean and tidy	169	7	65

Limitations

- Empirical results do not reflect most of benefits claimed by this algorithm (e.g. primitive action has no stochastic output in this sample domain)

- Needs a lot of human domain knowledge:
  - Relies on good domain-dependent choices in selecting a hierarchical formalization
  - Relies on proper design of the suggesters that suggests small set of plausible values from an infinite set of operator bindings.

Future Work for Paper / Reading

- Learning-based suggesters that reduce the human effort and improve over time
  - Beomjoon Kim and Leslie Pack Kaelbling and Tomas Lozano-Perez, **Adversarial actor-critic method for task and motion planning problems using planning experience**, In *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.
  - Rohan Chitnis, Dylan Hadfield-Menell, Abhishek Gupta, Siddharth Srivastava, E. Groshev, Christopher Lin, P. Abbeel, **Guided search for task and motion plans using learned heuristics**, In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.
- Learning-based symbolic operator for task level planning
  - Silver, Tom and Chitnis, Rohan and Tenenbaum, Josh and Kaelbling, Leslie Pack and Lozano-Perez, Tomas, **Learning Symbolic Operators for Task and Motion Planning**, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021
- Apply the replanning approach to uncertainty in outcomes, and extend the approach to apply to belief spaces in partially observable domains.
  - Leslie Pack Kaelbling and Tomás Lozano-Pérez, **Integrated Task and Motion Planning in Belief Space**, *International Journal of Robotics Research*, 2013
  - Garrett, Caelan R. and Paxton, Chris and Lozano-Perez, Tomas and Kaelbling, Leslie P. and Fox, Dieter, **Online Replanning in Belief Space for Partially Observable Task and Motion Problems**, In *International Conference on Robotics and Automation (ICRA)*, 2020.

Extended Readings

- A Review paper on TAMP
  - Garrett, Caelan Reed and Chitnis, Rohan and Holladay, Rachel and Kim, Beomjoon and Silver, Tom and Kaelbling, Leslie Pack and Lozano-Perez, Tomas, **Integrated Task and Motion Planning**, In *Annual review of control, robotics, and autonomous systems, volume 4*, 2021.
- Multi-modal motion planning: extend motion planning to handle constraints
  - Hauser K, Latombe JC, **Multi-modal Motion Planning in Non-expansive Spaces**, *International Journal of Robotics Research 29*, 2010
  - Marc Toussaint, Kelsey Allen, Kevin Smith, Joshua Tenenbaum, **Differentiable Physics and Stable Modes for Tool-Use and Manipulation Planning**, *Robotics: Science and Systems*, 2018

Summary

- This paper addresses the problem combining task and motion planning to solve complex manipulation tasks that requires sequential execution of multiple tasks with long time horizon.
- Such a combination requires the task planner to plan under continuous geometric variables and non-determinism from low-level motion planner
- Key insights:
  - Planning in the now: limiting the length of plans and reducing the amount of search required
  - Serializable planning steps: preventing failure plans from uncertain outputs
  - “Suggesters”: constraining task planning in limited suggested states
- They demonstrated these by analyzing a concrete example task from **Wash** domain. They showed how three insights work in this specific domain, and theoretically proved the correctness of the algorithm