

Towards a Plan Library for Household Robots

Armin Müller and Michael Beetz

Intelligent Autonomous Systems Group
Institut für Informatik
Technische Universität München

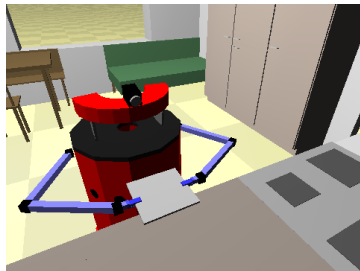
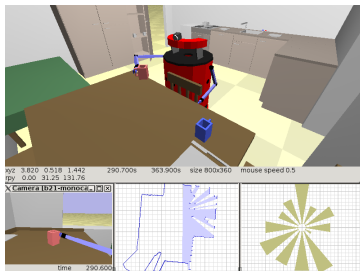
September 22, 2007

ICAPS 2007

3rd Workshop on
Planning and Plan Execution for Real-World Systems
Principles and Practices for Planning in Execution

The Household Robot Scenario

- represents everyday environment of human life
- unlike artificial environments
(unstructured, incomplete, uncertain knowledge)
- very hard for state-of-the-art robots
- robots have to perform everyday activities
(e.g. setting the table, cooking, cleaning)



Computational Problem

Given

- complete, inefficient plans
- set of transformation rules

Compute

- more **efficient** plans
- depending on **environment** and **dexterity** of the robot

Computational Problem

Given

- complete, inefficient plans
- set of transformation rules

Compute

- more **efficient** plans
- depending on **environment** and **dexterity** of the robot

Problems

- How to write and structure plans?
- How to build a library of **general** plans for everyday activities?
- How to implement **general** plan revisions?

Computational Problem

Given

- complete, inefficient plans
- set of transformation rules

Compute

- more **efficient** plans
- depending on **environment** and **dexterity** of the robot

Problems

- **How to write and structure plans?**
- **How to build a library of general plans for everyday activities?**
- **How to implement general plan revisions?**

Overview

This talk is about

- writing general, robust, transformable plans that robots can use in everyday situations
- our experience in the design of a plan library for a (simulated) autonomous robot

Outline

- 1 Motivation
- 2 Plan Library
- 3 Evaluation
- 4 Conclusion

Outline

- 1 Motivation
- 2 Plan Library**
- 3 Evaluation
- 4 Conclusion

Example Plan

entity-picked-up

inputs: entity

subgoals: (1) entity-found (2) at-location
(3) entity-gripped (4) entity-lifted

description: Search entity described by designator. If entity is already gripped, do nothing. Otherwise grip and lift entity at a suitable location.

Example Plan

entity-picked-up

inputs: entity

prepare: enclosing-container-opened

subgoals: (1) entity-found (2) at-location
(3) entity-gripped (4) entity-lifted

clean-up: container-closed

description: Search entity described by designator. If entity is already gripped, do nothing. Otherwise grip and lift entity at a suitable location.

Example Plan

entity-picked-up

inputs: entity

monitor: entity-lost-failure: when lifting entity

prepare: enclosing-container-opened

subgoals: (1) entity-found (2) at-location
(3) entity-gripped (4) entity-lifted

clean-up: container-closed

description: Search entity described by designator. If entity is already gripped, do nothing. Otherwise grip and lift entity at a suitable location.

Example Plan

entity-picked-up

inputs: entity

recover: (1) entity-lost-failure (2) grip-failure

monitor: entity-lost-failure: when lifting entity

prepare: enclosing-container-opened

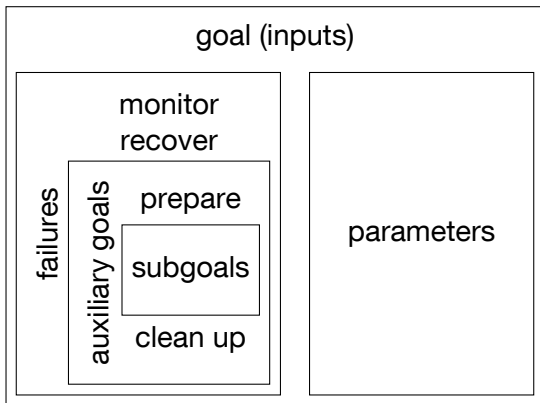
subgoals: (1) entity-found (2) at-location
(3) entity-gripped (4) entity-lifted

clean-up: container-closed

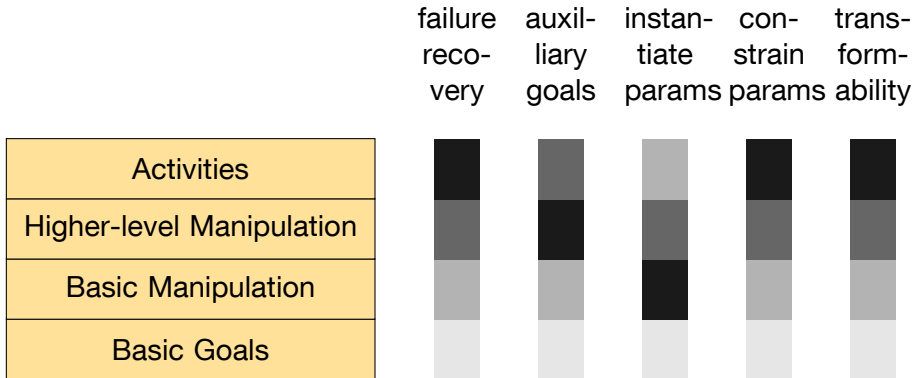
description: Search entity described by designator. If entity is already gripped, do nothing. Otherwise grip and lift entity at a suitable location.

Plan Structure

- general (not optimized for a special environment)
- reliable (monitor execution, detect and recover failures)
- transformable (understandable structure)



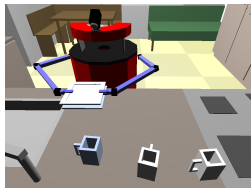
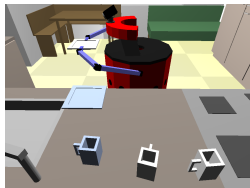
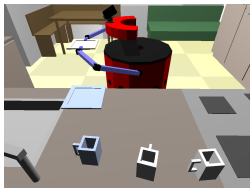
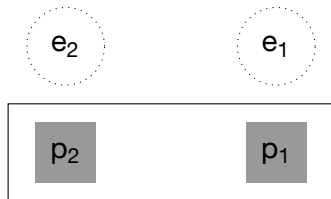
Plan Library Levels and their Characteristics



Combining Plan Steps

Hierarchical Interaction

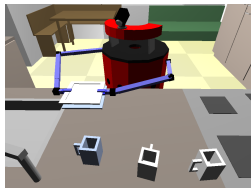
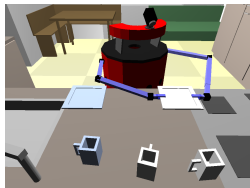
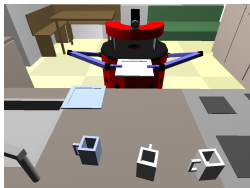
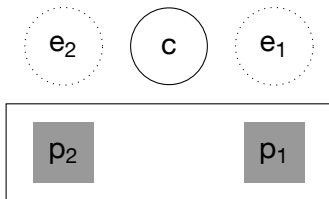
- lower-level instantiation leads to e_1 and e_2
- adding a constrain on an higher-level leads to c



Combining Plan Steps

Hierarchical Interaction

- lower-level instantiation leads to e_1 and e_2
- adding a constrain on an higher-level leads to c

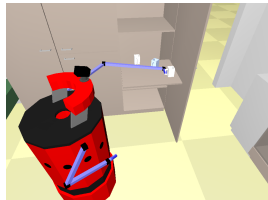
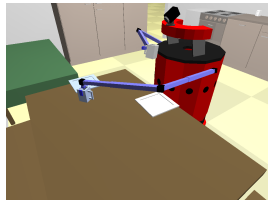


Combining Plan Steps

Sequential Interaction

When should auxiliary goals be executed?

- before/after each action: inefficient
- only when necessary: ?



Combining Plan Steps

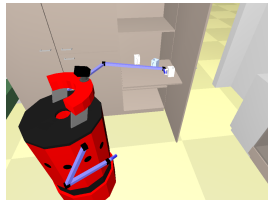
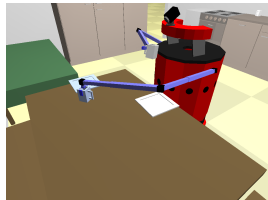
Sequential Interaction

When should auxiliary goals be executed?

- before/after each action: inefficient
- only when necessary: ?

Possible Solutions:

- skillfull programming
- plan transformation



Outline

- 1 Motivation
- 2 Plan Library
- 3 Evaluation**
- 4 Conclusion

Evaluation

- set the table uses six and boil pasta uses seven higher-level manipulation plans
- about eight basic manipulation plans
- hierarchy is 7–9 levels deep
- eight kind of failures are monitored
- in failure cases, 86% can be recovered successfully

Outline

- 1 Motivation
- 2 Plan Library
- 3 Evaluation
- 4 Conclusion**

Conclusion

- experience in developing a plan library for an autonomous household robot performing sophisticated everyday activities
- hierarchy of plans with different levels of abstraction
 - failure monitoring and recovery
 - determining parameters
 - auxiliary goals
 - transformability
- intricacies of combining plans