Probabilistic Planning for Robotics with ROSPlan

Gerard Canal, Research Assistant

KING'S College LONDON

Planning in Robotics

- Uncertain domains → Chance of failure!

Planning in Robotics

- Uncertain domains → Chance of failure!
- Two approaches:

Planning in Robotics

- Uncertain domains → Chance of failure!
- Two approaches:
  - Deterministic planning + replanning on failure

Planning in Robotics

- Uncertain domains → Chance of failure!
- Two approaches:
  - Deterministic planning + replanning on failure
  - Probabilistic planning optimizing success probability

Planning in Robotics

- Probabilistically interesting problems

Planning in Robotics

- Probabilistically interesting problems:
  - Multiple plans with different rewards based on success probabilities

Planning in Robotics

- Probabilistically interesting problems:
  - Multiple plans with different rewards based on success probabilities
  - Possible dead-ends in the state space

Probabilistic Planning Languages

- Relational Dynamic Influence Diagram Language (RDDL)

Probabilistic Planning Languages

- Relational Dynamic Influence Diagram Language (RDDL)
  - Used in the last IPPC editions

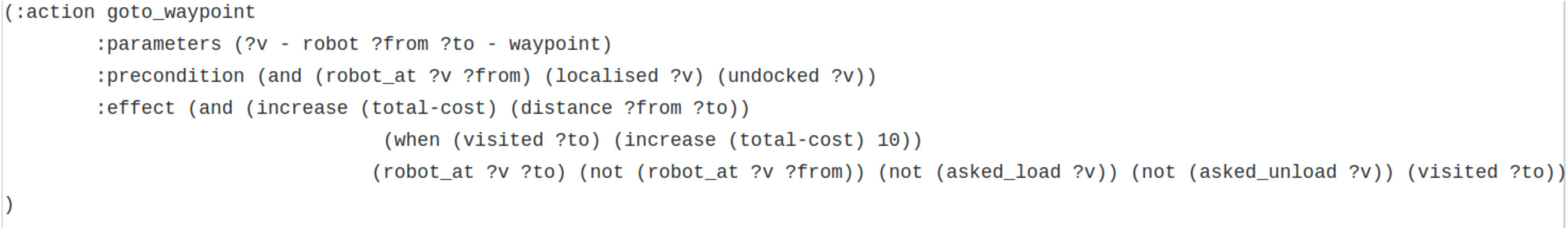
Probabilistic Planning Languages

- Relational Dynamic Influence Diagram Language (RDDL)
  - Used in the last IPPC editions
  - Most state-of-the-art probabilistic planning use it

Probabilistic Planning Languages

- Relational Dynamic Influence Diagram Language (RDDL)
  - Used in the last IPPC editions
  - Most state-of-the-art probabilistic planning use it
- Probabilistic Planning Domain Description language (PPDDL)
  - Extension of PDDL to probabilistic outcomes

Planning Languages: PDDL



```
(:action goto_waypoint
        :parameters (?v - robot ?from ?to - waypoint)
        :precondition (and (robot_at ?v ?from) (localised ?v) (undocked ?v))
        :effect (and (increase (total-cost) (distance ?from ?to))
                                 (when (visited ?to) (increase (total-cost) 10))
                                (robot_at ?v ?to) (not (robot_at ?v ?from)) (not (asked_load ?v)) (not (asked_unload ?v)) (visited ?to))
)
```

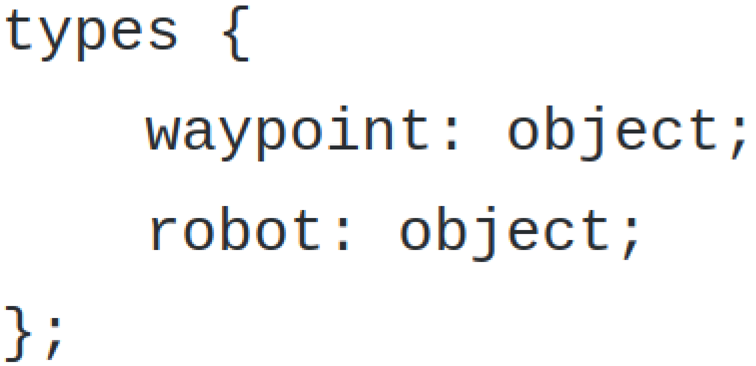
Planning Languages: RDDL

```
    // Action fluents
    goto_waypoint(robot, waypoint, waypoint): { action-fluent, bool, default = false }; // robot from to
    localise(robot): { action-fluent, bool, default = false };
    dock(robot, waypoint): { action-fluent, bool, default = false };
    undock(robot, waypoint): { action-fluent, bool, default = false };
};

cpfs {

    robot_at'(?r, ?w) = if (exists_{?w1: waypoint} (goto_waypoint(?r, ?w1, ?w))) then true
                        else if (exists_{?w1: waypoint} (goto_waypoint(?r, ?w, ?w1))) then false
                        else robot_at(?r, ?w);
```

RDDL Description

- Sections:
  - Types
  - pvariables
  - cpfs
  - rewards
  - action_preconditions

Types



```
types {
    waypoint: object;
    robot: object;
};
```

pvariables

```
pvariables {
        DOCK_AT(waypoint): { non-fluent, bool, default = false};
        robot_at(robot, waypoint): { state-fluent, bool, default = false };
        dock(robot, waypoint): { action-fluent, bool, default = false };
}
```

cpfs

```
cpfs {

  robot_at'(?r, ?w) = if (exists_{?w1: waypoint} (goto_waypoint(?r, ?w1, ?
w))) then true



  undocked'(?r) = undocked(?r) ^ ~(exists_{?w: waypoint} (dock(?r, ?w))) |
docked(?r) ^ (exists_{?w: waypoint} (undock(?r, ?w)));



   somebody_at'(?w) = Bernoulli(0.75);

}
```

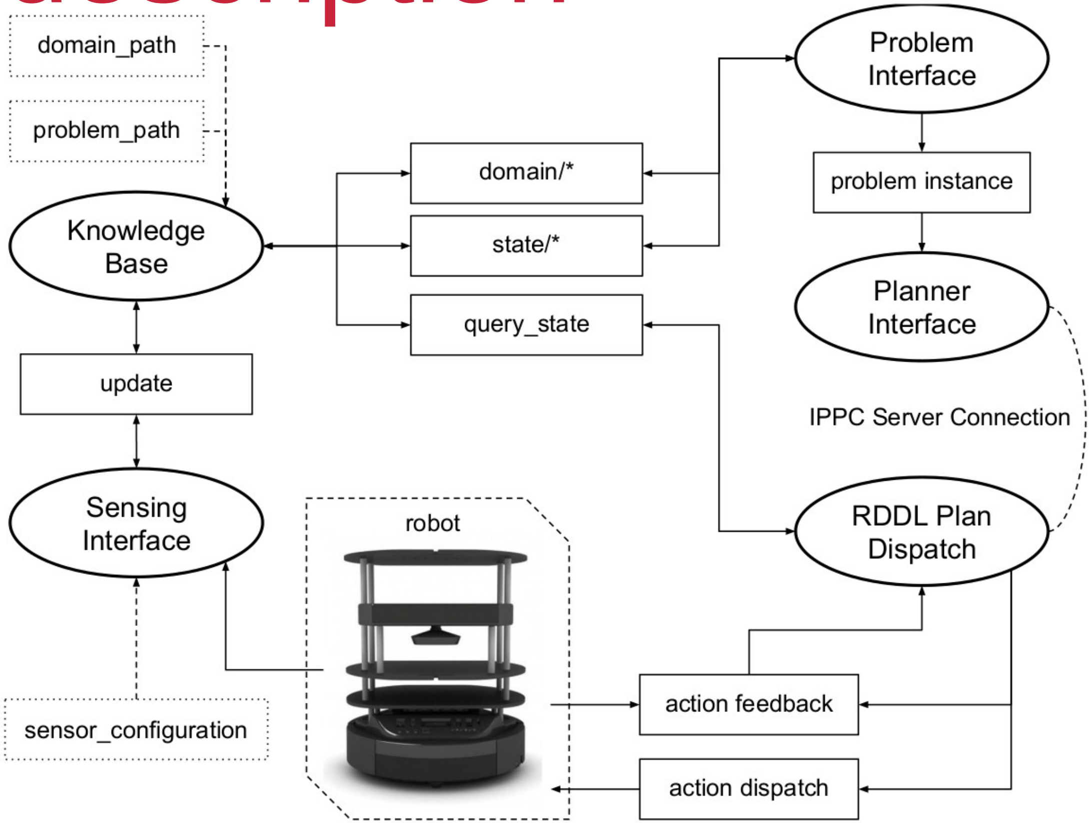
reward

```
reward = if (exists_{?r: robot}[robot_at(?r, goal)]) then 500 else -5;
```

action_preconditions

```
action-preconditions {

  // A robot must be undocked, localised and in a position to move to
another

 forall_{?r: robot, ?wf: waypoint, ?wt: waypoint} [goto_waypoint(?r, ?wf,
?wt) => (robot_at(?r, ?wf) ^ localised(?r) ^ undocked(?r))];
```

System description



Knowledge Base

- Stores the current model of the environment

Knowledge Base

- Stores the current model of the environment
- RDDL integrated using the **same** interface.

Knowledge Base

- Stores the current model of the environment
- RDDL integrated using the **same** interface.
  - Preserves compatibility

Knowledge Base

- Stores the current model of the environment
- RDDL integrated using the **same** interface.
    - Preserves compatibility
    - Interchange of PDDL and RDDL KBs

RDDL Knowledge Base

- RDDL is translated to PDDL-like structures

RDDL Knowledge Base

- RDDL is translated to PDDL-like structures
  - Not everything can be converted

RDDL Knowledge Base

- RDDL is translated to PDDL-like structures
  - Not everything can be converted
- action-fluents are mapped to PDDL operators

RDDL Knowledge Base

- RDDL is translated to PDDL-like structures
  - Not everything can be converted
- action-fluents are mapped to PDDL operators
- state-action constraints are preconditions:

RDDL Knowledge Base

- RDDL is translated to PDDL-like structures
  - Not everything can be converted
- action-fluents are mapped to PDDL operators
- state-action constraints are preconditions:
  - action-fluent → formula
  - The formula is encoded as a precondition

RDDL Knowledge Base

- Action effects are obtained from the conditional probability formulas (cpfs)

RDDL Knowledge Base

- Action effects are obtained from the conditional probability formulas (cpfs)
  - Describe how a fluent changes based on the current state

RDDL Knowledge Base

- Action effects are obtained from the conditional probability formulas (cpfs)
  - Describe how a fluent changes based on the current state
- Probabilistic effects are considered

RDDL Knowledge Base

- Action effects are obtained from the conditional probability formulas (cpfs)
  - Describe how a fluent changes based on the current state
- Probabilistic effects are considered
- Exogenous effects too!

Problem generation

- Problems can be generated in RDDL

Problem generation

- Problems can be generated in RDDL
- KB interfaces are the same

Problem generation

- Problems can be generated in RDDL
- KB interfaces are the same
  - We can generate problems in different languages from the same Knowledge Base model!

Online planning and execution

- Non-deterministic planners and MDP solvers are usually online

Online planning and execution

- Non-deterministic planners and MDP solvers are usually online
  - Plan-as-you-go approach

Online planning and execution

- Non-deterministic planners and MDP solvers are usually online
  - Plan-as-you-go approach
  - Convenient for robotics, no need for replanning

Online planning and execution

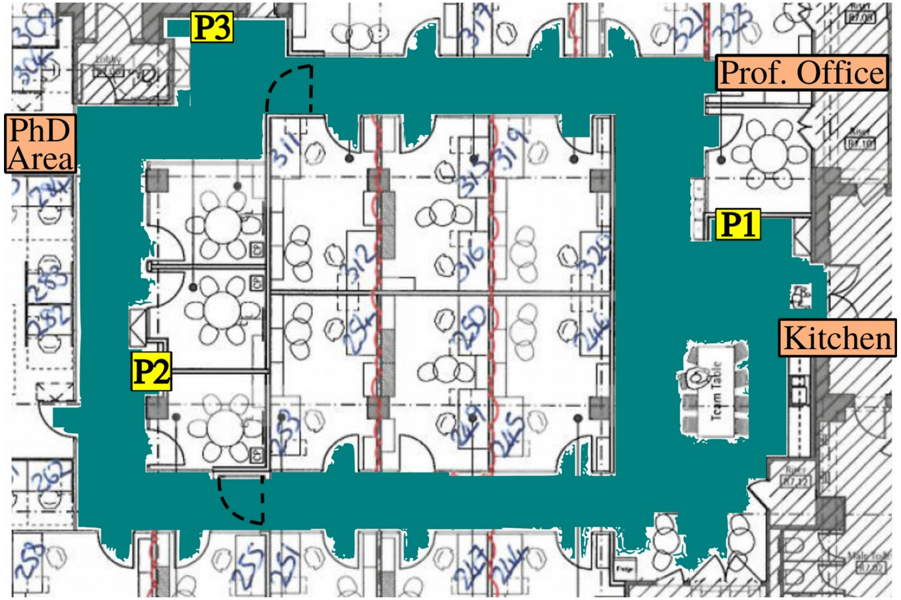
- Non-deterministic planners and MDP solvers are usually online
  - Plan-as-you-go approach
  - Convenient for robotics, no need for replanning
- Need for a different dispatcher!

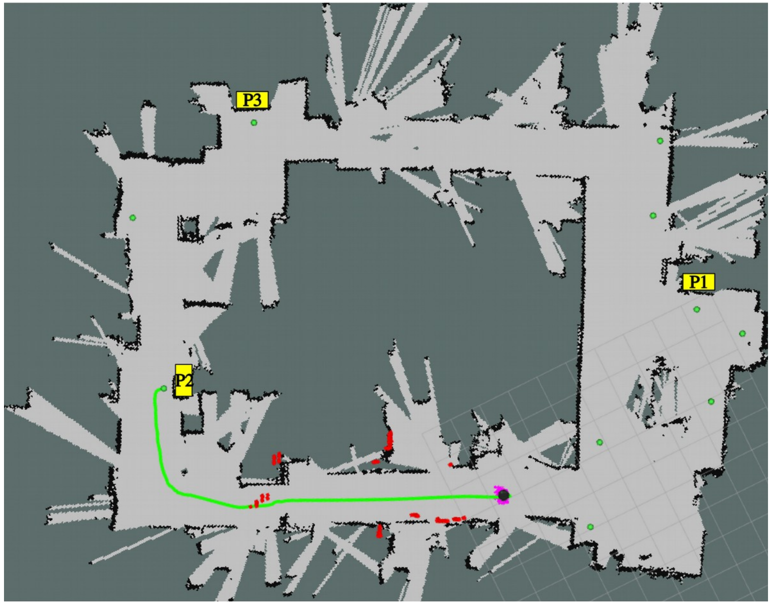
Non-deterministic effects (Reminder)

- Deterministic effects always happen (if action succeeds)
- Non-deterministic effects create divergent paths
  - Need to update KB accordingly!
  - How to detect what happened?
    - SENSORS!

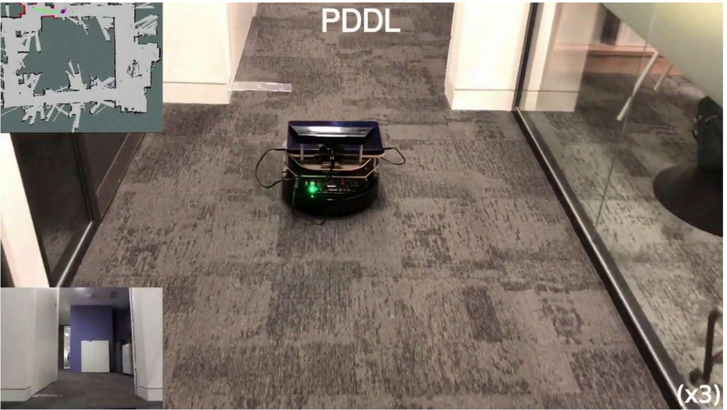
Experiments

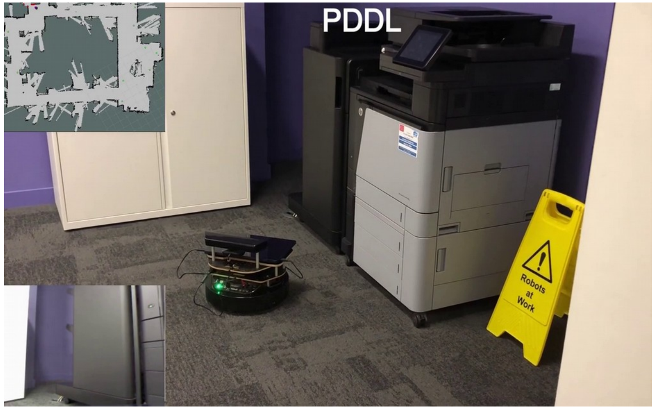
- Example scenario: print fetching



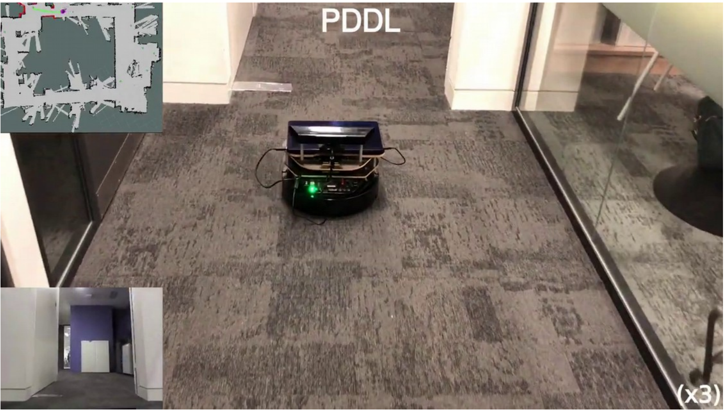


Experiments

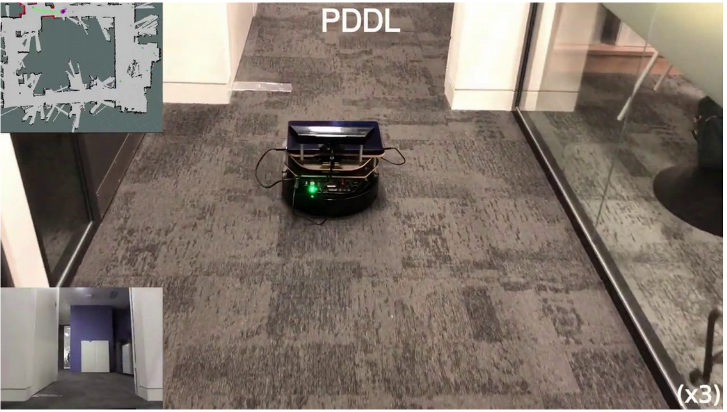


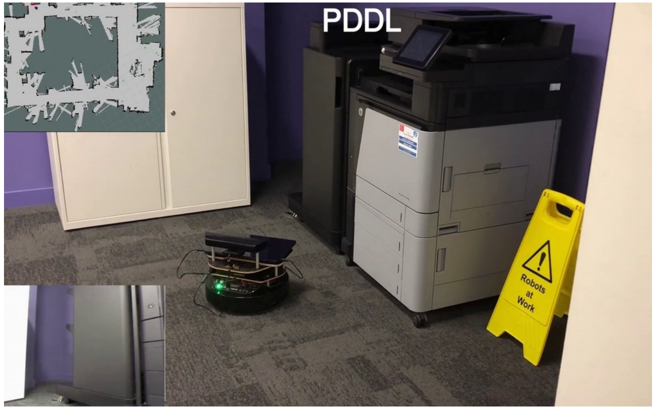


Experiments

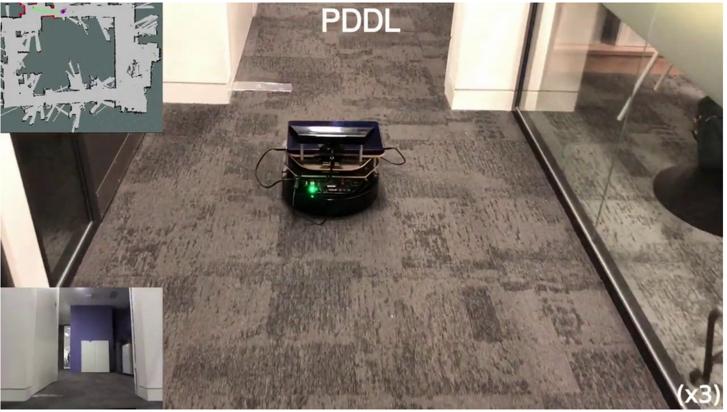


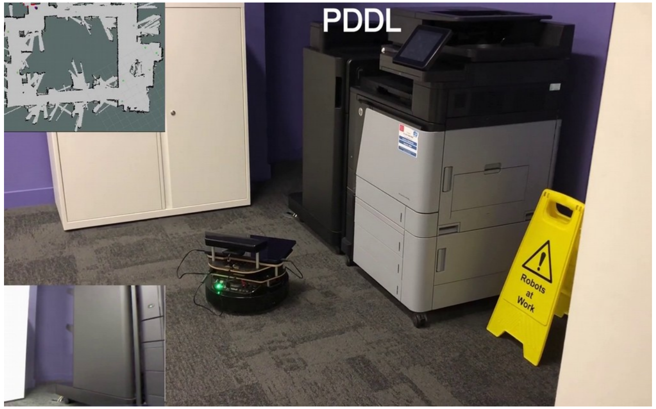
Experiments



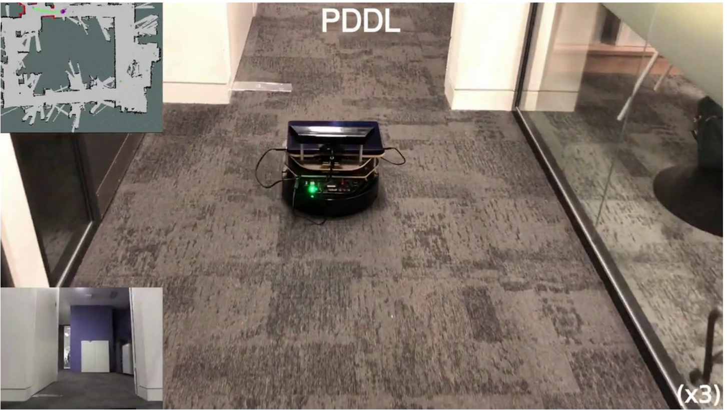


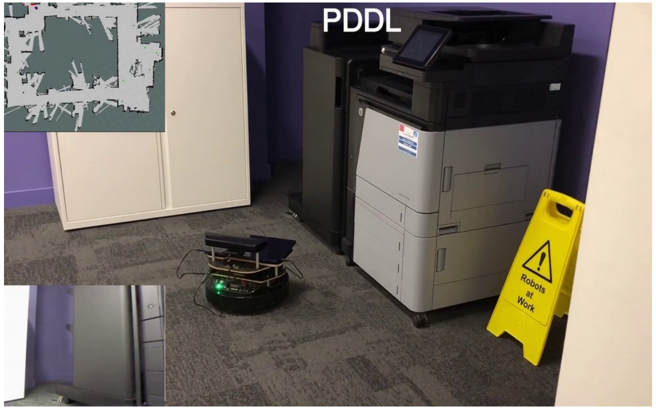
Experiments





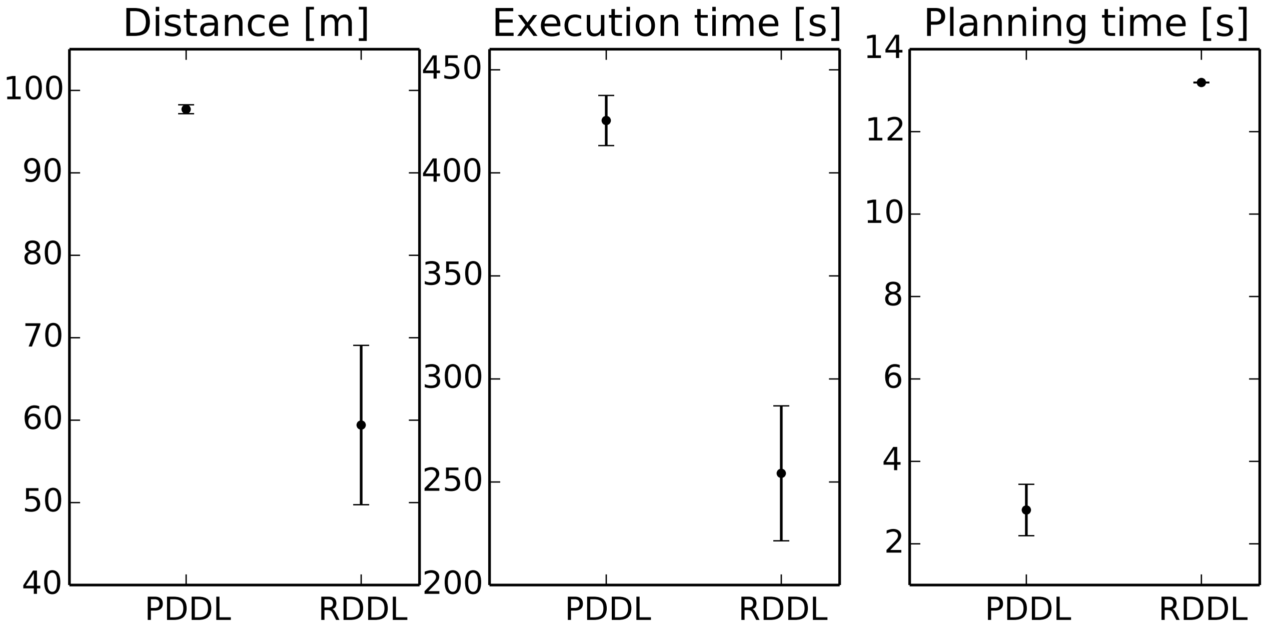
Experiments





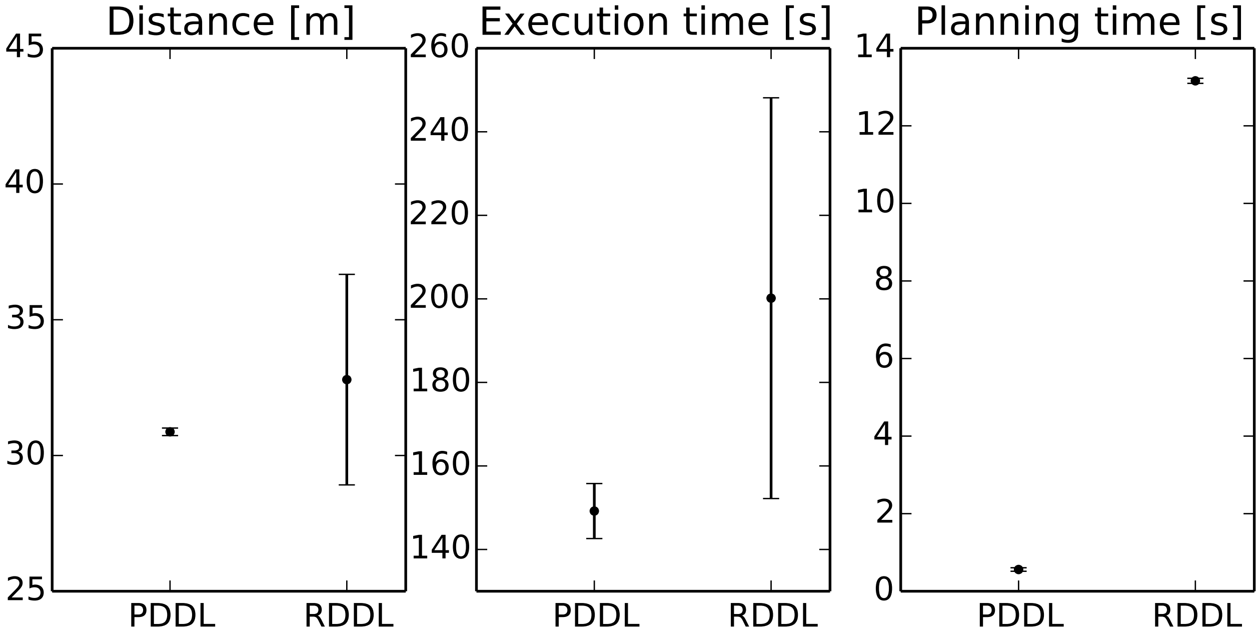
Results: experiment 1

- All printers free
- Person in closest printer



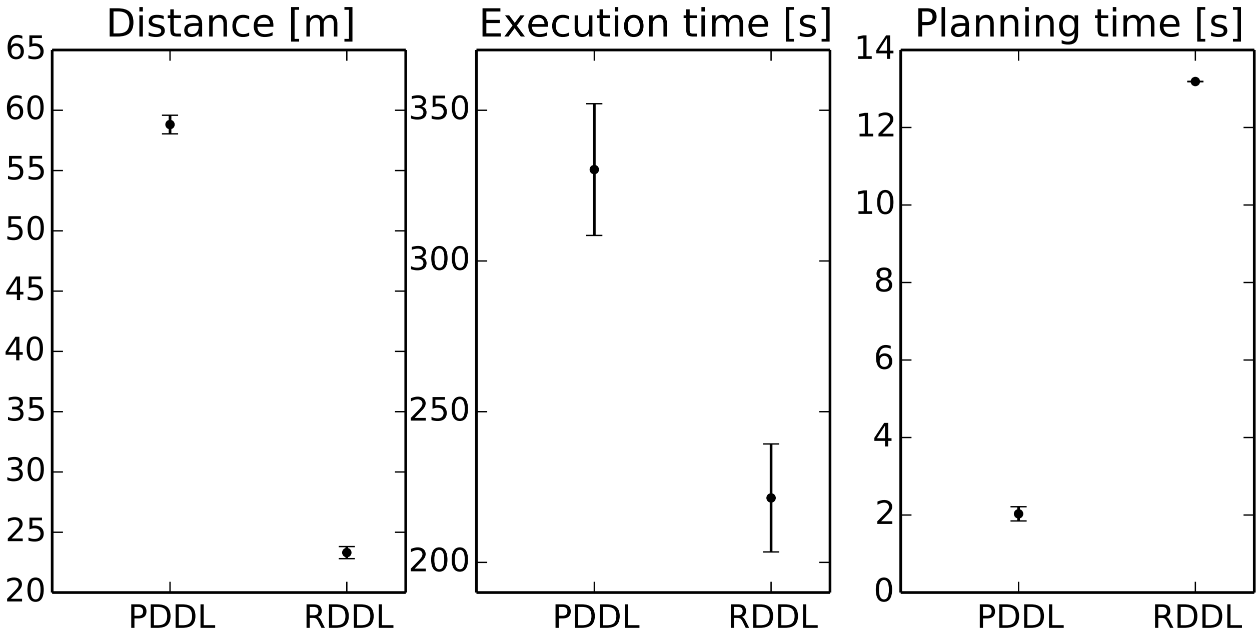
Results: experiment 2

- Closest printer busy
- People everywhere



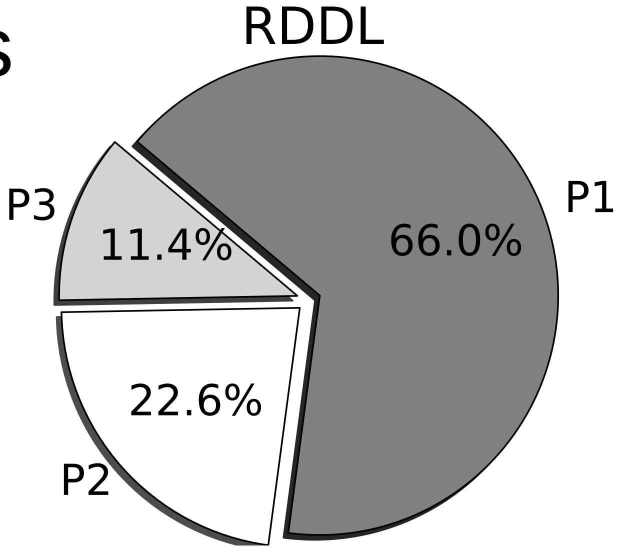
Results: experiment 3

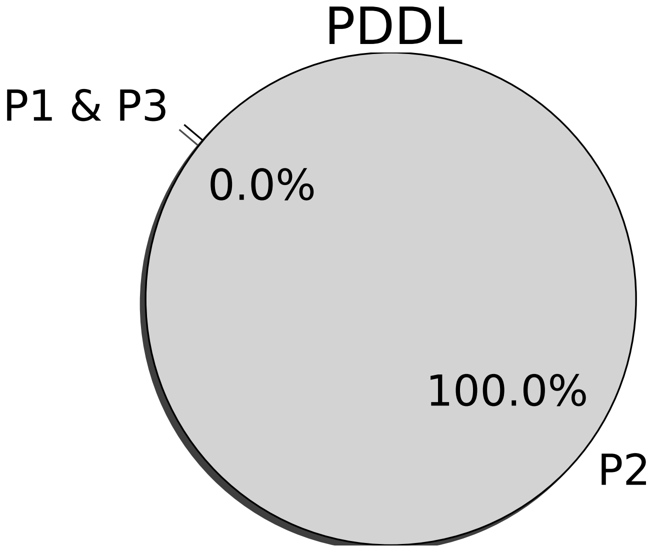
- Free printer without people
- Others busy



Results: Simulation

- 500 executions
- Check first chosen printer





Video demo

Conclusions

- Disclaimer: Not intended to decide which planning approach is better.

Conclusions

- Disclaimer: Not intended to decide which planning approach is better.
- Integration of RDDL into existing PDDL-based ROSPlan Knowledge Base.

Conclusions

- Disclaimer: Not intended to decide which planning approach is better.
- Integration of RDDL into existing PDDL-based ROSPlan Knowledge Base.
- Possible combination of probabilistic and deterministic approaches together, resulting in an hybrid system.

Thank you for your attention!

Questions are welcomed!