Knowledge-Level Planning for Task-Based Human-Robot Interaction

Ron Petrick
School of Informatics
University of Edinburgh
Edinburgh, Scotland, United Kingdom
rpetrick@inf.ed.ac.uk

IPAB Workshop
Institute of Perception, Action, and Behaviour
University of Edinburgh
29 January 2015
Two people walk into a bar...

Two people, A and B, each individually approach a bartender.

Bartender (to A): How can I help you?
Person A: A pint of cider, please.

Person C approaches the bartender and attracts his attention by gesturing.

Bartender (to C): How can I help you?
Person C: I’d like a pint of bitter.
Bartender: (Serves C)
Bartender (to B): What will you have?
Person B: A glass of red wine.
Bartender: (Serves B)
Bartender: (Serves A)
Two people walk into another bar...

Two people, A and B, each individually approach a bartender.

Bartender (to A): How can I help you?
Person A: A pint of cider, please.

Person C approaches the bartender and attracts his attention by gesturing.

Bartender (to C): Just a moment please.
Bartender: (Serves A)
Bartender (to B): What will you have?
Person B: A glass of red wine.
Bartender: (Serves B)
Bartender (to C): Thanks for waiting. How can I help you?
Person C: I’d like a pint of bitter.
Bartender: (Serves C)
Two interactions

Two people, A and B, each individually approach a bartender

Bartender (to A): How can I help you? Bartender (to A): How can I help you?
Person A: A pint of cider, please. Person A: A pint of cider, please.

Person C approaches the bartender and attracts his attention by gesturing

Bartender (to C): How can I help you? Bartender (to C): Wait a moment please
Person C: I’d like a pint of bitter. Bartender: (Serves A)
Bartender: (Serves C)
Bartender (to B): What will you have? Person B: A pint of Guinness.
Person B: A pint of Guinness. Bartender: (Serves B)
Bartender: (Serves B)
Bartender (to C): Thanks for waiting. Bartender (to C): Thanks for waiting.
How can I help you?
Person C: I’d like a pint of bitter. Bartender: (Serves C)
Bartender: (Serves C)

• Both interactions result in the customers achieving their task goals.
• The first interaction is shorter.
• The second interaction can be seen to be more socially appropriate.
Meet the bartender: JAMES

• Part of the JAMES Project (http://james-project.eu/), funded by the European Commission, exploring social interaction in robotics domains.
Why social interaction?

• **Successful task interaction often relies on social interaction.**
  – May be several ways to achieve a task-based goal.
  – Appropriate social behaviour can lead to higher participant satisfaction.

• **Social interaction can be seen as an instance of joint action.**
  – Involves coordination of participant actions.
  – Inherently multimodal: speech, gesture, gaze, expression, etc.

• **Social interaction is often multi-party, dynamic, short-horizon.**
  – In contrast to one-on-one, companion-style relationships.
  – Interactions are often “one shot”; may not have an opportunity to recover from a poor interaction.
Task-based social interaction

• Robot bartender must respond to user requests in a dynamic setting with multiple users and short interactions in German or English.

• Interactions incorporate both **task-based aspects** (e.g., ordering and serving drinks) and **social aspects** (e.g., managing multiple interactions).

• Supported activities include
  - Asking customers for drink orders
  - Handing over drinks
  - Tracking the order people arrive at the bar but not...
  - Pouring drinks, handling money, small talk, ...

• How important is **social interaction**?
Robot hardware

Physical hardware

Simulated hardware

Credit: A. Gaschler and M. Giuliani, fortiss GmbH
Vision system

Full system

Kinect-only

Credit: M. Pateraki and M. Sigalas, FORTH

• See (Pateraki et al. 2013) for more information.
A multidisciplinary architecture

- An important aspect of the JAMES research is the collection and analysis of data collected from real bars investigating how human customers interact with human bartenders (Huth 2011; Loth et al. 2013).
High-level action selection in JAMES

- What action should the robot perform next?
- We use **automated planning** techniques from the symbolic artificial intelligence community, which are good at building goal-directed plans of action under many challenging conditions, given a formal description of a domain.
- **Goal:** build plans for serving all agents seeking attention in the bar (Loth et al. 2013; Foster et al. 2013).
High-level action selection

1. Greet the customer,
2. Ask the customer for a drink,
3. Acknowledge the drink order,
4. Pickup the correct bottle,
5. Serve the customer,
6. Close the transaction.
From dialogue and interaction to planning

- A key modality in human social interaction is natural language dialogue.
- Parallels to planning: a speaker tries to change the mental state of the hearer by applying actions that correspond to the utterance of words.
- The link between natural language and planning has a long tradition, e.g., Perrault and Allen (1980); Appelt (1985); Clark (1996); Stone (2000); Litman and Allen (1987); Cohen and Levesque (1990); Grosz and Sidner (1990), ...
- Recent work has tended to separate task planning from other types of natural language planning, which use more specialised approaches, e.g., finite state machines, information state (e.g., TrindiKit (Larsson and Traum 2000)), rule-based approaches to speech act theories, dialogue games, ...
- **Challenge**: replace the behaviour of a traditional interaction/dialogue manager with a general-purpose AI planner (Petrick and Foster 2013).
Automated AI planning

• Automated AI **planning** techniques are good at building goal-directed plans of action under many challenging conditions, given a suitable description of a domain.

• A **planning problem** consists of:
  1. A representation of the properties and objects in the world and/or the agent’s knowledge, usually described in a logical language,
  2. A set of state transforming actions,
  3. A description of the initial world/knowledge state,
  4. A set of goal conditions to be achieved.

• A **plan** is a sequence of actions that when applied to the initial state transforms the state in such a way that the resulting state satisfies the goal conditions.
Target interaction

Two people, A and B, each individually approach a bartender.

Bartender (to A):
How can I help you? Sensing action
Person A: A pint of cider, please.

Person C approaches the bartender and attracts his attention.

Bartender (nods at A, then to C):
Just a moment please. Social action
(Serves A) Physical action

Bartender: What will you have? Sensing action
Person B: A glass of red wine.

Bartender (nods at B):
(Serves B) Physical action

Bartender (to C):
Thanks for waiting. Social action
How can I help you? Sensing action

Person C: I’d like a pint of bitter.

Bartender (nods at C):
(Serves C) Physical action
Planning in JAMES

• Input (fusion): sensor information from vision and speech.
  Output (fission): actions are postprocessed to generate arm motions, head behaviour, and speech.

• Domain includes:
  – Physical actions (e.g., handing over a drink),
  – Information-gathering (sensing) actions (e.g., asking a customer for a drink order) → often correspond to dialogue acts, and
  – Social behaviour (e.g., acknowledgements, thanking a customer).

• **Goal**: build plans to transact with all customers (agents) seeking attention at the bar (Petrick and Foster 2013; Foster et al. 2013).
PKS: Planning with Knowledge and Sensing

• Our approach: treat the problem as **planning with incomplete information and sensing**.

• Plans are generated using **PKS** (Petrick and Bacchus 2002, 2004), a knowledge-level contingent planner that builds plans based on the planner’s knowledge state.

• PKS uses an extended STRIPS-style representation, based collection of five databases, each of which is restricted to a particular type of knowledge: $K_f$, $K_v$, $K_w$, $K_x$, $LCW$.

• The contents of the databases ($DB$) have a fixed formal translation to formulae in a modal logic of knowledge which formally defines the planner’s knowledge state ($KB$).

• Actions are defined in terms of the changes they make to the planner’s knowledge state (i.e., the databases), rather than the world state.

• Plans are generated by forward search: actions update $DB \Rightarrow$ update $KB$. 
Representing knowledge in PKS

• $K_f$: knowledge of positive and negative facts (but not closed world!)

\[ p(c) \quad \neg q(b, c) \quad f(a) = c \quad g(b, c) \neq d \]

• $K_w$: knowledge of binary sensing effects

\[ \phi \in K_w : \text{the planner knows whether } \phi \]

• $K_v$: knowledge of function values, multi-valued sensing effects

\[ f \in K_v : \text{the planner knows the value of } f \]

• $K_x$: exclusive-or knowledge

\[ (\ell_1|\ell_2|\ldots|\ell_n) \in K_x : \text{exactly one of the } \ell_i \text{ must be true} \]

• $LCW$: local closed world information (Etzioni et al. 1994)
Representing actions in PKS

action ask-drink(?a : agent, ?g : group)
  preconds: K(inTrans = ?g) & K(inGroup(?a) = ?g) & K(!ordered(?a)) & K(!otherAttnReq)
effects: add(Kf,ordered(?a)),
        add(Kv,request(?a))

action ack-order(?a : agent, ?g : group)
  preconds: K(inTrans = ?g) & K(inGroup(?a) = ?g) & K(ordered(?a)) & K(!ackOrder(?a)) & K(!otherAttnReq)
effects: add(Kf,ackOrder(?a))

• Actions capture the changes they make to the knowledge state.
• New knowledge states are computed by forward chaining.
• Plans are built by chaining actions together using search.
Initial states and state updates

- Many aspects of the operating environment are dynamic and cannot be determined a priori: agents in the scene, agents seeking attention, initial utterances, etc.

- A **state manager** supports the planner by turning the **continuous** stream of noisy low-level sensor inputs into a **discrete** state representation (Foster 2014).

- Sensors: linguistic interpreter, vision, robot arms, speech synthesiser, talking head.

- State properties reported to the planner:
  - Basic agent properties: location, face/torso orientation,
  - Inferred agent properties: social state, drink requests,
  - System output state: bad automatic speech recognition.
Domain 1: a social bartender

• Actions
  
  ```
  greet(?a,?g)  
  ask-drink(?a,?g)  
  ask-drink-next(?a,?g)  
  serve(?a,?d,?g)  
  bye(?a,?g)  
  wait(?a,?g)  
  ack-order(?a,?g)  
  ack-wait(?a,?g)  
  ack-thanks(?a,?g)  
  inform-drinklist(?a,?t)  
  ```
  
  greet agent ?a in group ?g  
  ask agent ?a in group ?g for a drink order  
  ask the next agent ?a in group ?g for a drink order  
  serve drink ?d to agent ?a in group ?g  
  end an interaction with agent ?a in group ?g  
  tell agent ?a in group ?g to wait  
  acknowledge the order of agent ?a in group ?g  
  thank agent ?a in group ?g for waiting  
  acknowledge agent ?a’s thanks  
  inform agent ?a of the available drinks of type ?t

• Properties
  
  ```
  seeksAttn(?a)  
  visible(?a)  
  inGroup(?a) = ?g  
  inTrans = ?g  
  request(?a) = ?d  
  ```
  
  agent ?a seeks attention  
  agent ?a is visible  
  agent ?a is in group ?g  
  the robot is interacting with group ?g  
  agent ?a has requested drink  
  ...

⇒ The social bartender domain is described symbolically for the planner, inspired by data collected from human studies in real bars (Huth 2011).
A plan for serving a single customer

\[
greet(a1, g1), \\
ask\text{-drink}(a1, g1), \\
ack\text{-order}(a1, g1), \\
serve(a1, \text{request}(a1), g1), \\
bye(a1, g1). \\
\]

• Simplest possible plan in the single customer case.
• Plans are built in response to customers seeking attention in the bar.
• Represent best-case scenarios based on current state information.
Example: asking a customer for a drink

Plan step

`ask-drink(A1)`

Multimodal output specification

State update

`add(Kf, request(A1)=water))`

Parsed speech input

```
<lf>
  <node id="c1:drink" pred="water" mood="dcl" num="sg" />
</lf>
```

See (Petrick et al. 2012) for more information.
Example: serving a drink to a customer

See (Petrick et al. 2012) for more information.
A plan for two customers in two groups

wait(a2,g2),
greet(a1,g1),
ask-drink(a1,g1),
ack-order(a1,g1),
serve(a1/request(a1),g1),
bye(a1,g1),
ack-wait(a2,g2),
ask-drink(a2,g2),
ack-order(a2,g2),
serve(a2/request(a2),g2),
bye(a2,g2).

[Tell group g2 to wait]
[Greet group g1]
[Ask a1 for drink order]
[Acknowledge a1’s order]
[Give the drink to a1]
[End g1’s transaction]
[Thank g2 for waiting]
[Ask a2 for drink order]
[Acknowledge a2’s order]
[Give the drink to a2]
[End g2’s transaction]

• If a new customer arrives while the bartender is occupied, it nods at them and serves them later.
A plan for two customers in one group

greet(a1,g1),
ask-drink(a1,g1),
ack-order(a1,g1),
ask-drink-next(a2,g1),
ack-order(a2,g1),
serve(a1,request(a1),g1),
serve(a2,request(a2),g1),
bye(a2,g1).

[Greet group g1]
[Ask a1 for drink order]
[Acknowledge a1’s order]
[Ask a2 for drink order]
[Acknowledge a2’s order]
[Give the drink to a1]
[Give the drink to a2]
[End g1’s transaction]

• When groups are detected, all individuals in a group are asked for their drink orders before any drinks to the group are served.

• Note: we can also support “round-buying” behaviour by making small changes to the planning domain.
A single customer conditional plan

```
greet(a1,g1), 
ask-drink(a1,g1), 
branch(request(a1)) 
  K(request(a1)=juice): 
    ... 
    serve(a1,juice,g1) 
  K(request(a1)=water): 
    ... 
    serve(a1,water,g1) 
  K(request(a1)=beer): 
    ... 
    serve(a1,beer,g1) 
bye(a1,g1).
```

[Greet agent a1] [Ask a1 for drink order] [Form branching plan] [If order is juice] [If order is water] [If order is beer] [Serve juice to a1] [Serve water to a1] [Serve beer to a1] [End the transaction]

- Branches let the planner consider order-specific actions/subdialogues.
A more complex interaction

Three customers:
A1 and A2 in group G1
A3 is alone (singleton group G2)
Bartender serves members of G1 in sequence, then deals with G2.

Other social behaviour:
• First-come/first-served ordering
• All orders are acknowledged immediately
• If a new customer arrives while the bartender is occupied, it nods at them and serves them later

Social behaviour is based on the observation of bartenders in real bars (Huth et al., 2012); see Foster et al. (2013) for details on the planning domain.

wait(A3, G1)
greet(A1, G1)
ask-drink(A1, G1)
ack-order(A1, G1)
ask-drink(A2, G1)
ack-order(A2, G1)
serve(A1, request(A1), G1)
serve(A2, request(A2), G2)
bye(A2, G1)
ack-wait(A3, G2)
ask-drink(A3, G2)
ack-order(A3, G2)
serve(A3, request(A3), G3)
bye(A3, G2)
Tell G2 to wait (with a nod)
Greet group G1
Ask A1 for drink order
Acknowledge A1’s order
Ask A2 for drink order
Acknowledge A2’s order
Give the drink to A1
Give the drink to A2
End G1’s transaction
Acknowledge G2’s wait
Ask A3 for drink order
Acknowledge A3’s order
Give the drink to A3
End G2’s transaction
Replanning when things go wrong

- Interactions are continually monitored to detect problems that may trigger replanning.

- Low-confidence speech recognition / timeouts

  ...  
  ask-drink(a1)  
  ???  
  [Replan]  
  not-understand(a1)  
  ask-drink(a1)  
  ...  

  [Ask a1 for drink order]  
  [a1 was not understood]  
  [Replan]  
  [Alert a1 not understood]  
  [Ask a1 again for drink order]  
  [Continue with old plan]  

- Overanswering

  greet(a1)  
  ???  
  [Replan]  
  serve(a1, request(a1))  
  bye(a1)  
  ...  

  [Greet a1]  
  [a1 says “I’d like a beer”]  
  [Replan]  
  [Serve a1 their drink]  
  [End the transaction with a1]
JAMES interaction video

http://youtu.be/8k7Pd-CbbhE
http://james-project.eu/
Experiment 1 and results

• System tested with 2 customers at a time in a drink ordering scenario (31 participants × 3 interactions each): 95% success rate on delivering correct drinks. More details in (Foster et al. 2012).

• Planning time is typically quite short, which doesn’t negatively impact the system’s reaction time (e.g., plans for 3 customers require 17 steps and <0.1s generation time).
  – Anything less than 2s is usually okay.
  – Robot motions are relatively slow which offers future opportunities for parallelising planning with other robot activities.
  – Frequent replanning in this domain.
Domain 2: a task-only bartender

• Actions
  
greet(?a,?g)  greet agent ?a in group ?g
ask-drink(?a)  ask agent ?a for a drink order
ask-drink-next(?a,?g)  ask the next agent ?a in group ?g for a drink order
serve(?a,?d)  serve drink ?d to agent ?a
bye(?a,?g)  end an interaction with agent ?a in group ?g
wait(?a,?g)  tell agent ?a in group ?g to wait
ack-order(?a,?g)  acknowledge the order of agent ?a in group ?g
ack-wait(?a,?g)  thank agent ?a in group ?g for waiting
ack-thanks(?a,?g)  acknowledge agent ?a’s thanks
inform-drinklist(?a,?t)  inform agent ?a of the available drinks of type ?t

• Properties
  
seeksAttn(?a)  agent ?a seeks attention
visible(?a)  agent ?a is visible
inGroup(?a) = ?g  agent ?a is in group ?g
inTrans = ?g  the robot is interacting with group ?g
request(?a) = ?d  agent ?a has requested drink
...

⇒ A task-only version of the bartender domain is formed by disabling certain actions and properties from the social domain.
A plan for serving a single customer

ask-drink(a1), serve(a1, request(a1)).

[Ask a1 for drink order] [Give the drink to a1]

• The decision to interact with a customer is motivated by the visibility of the customer, not by the fact that the customer is seeking attention.

• No notion of groups.

• No social actions: plans are purely task based.
A plan for three customers

ask-drink(a1), [Ask a1 for drink order]
ask-drink(a2), [Ask a2 for drink order]
serve(a2, request(a2)), [Give the drink to a2]
ask-drink(a3), [Ask a3 for drink order]
serve(a3, request(a3)), [Give the drink to a3]
serve(a1, request(a1)). [Give the drink to a1]

• In theory, the planner may generate any interleaving of the three single agent plans.
Experiment 2: social vs. task-only

• More complex scenario (3 customers at a time) involving group detection and a comparison of a task-only version of the domain versus the full social domain (Giuliani et al. 2013).

• 40 participants (28 male) drawn from university departments outside the robotics group.

• Each participant could choose to interact with the robot in English (14) or German (26).

• Before the experiment, participants were told they would order a drink from a robot bartender.

• Without seeing the robot, participants were asked to rate their expectations in a questionnaire (based on standard GODSPEED).

• Participants were then shown the robot and told to order a drink from a list of possible drinks. No other instructions were given.

• Following the interaction, participants completed the same questionnaire as at the start of the experiment.
Experiment 2: results

- **Objective results**
  - Two factors had significant impact on the objective results: the choice of interaction domain and the participants' gender.
  - The first drink was served more quickly with the social version, which also took more system turns but not more time.
  - Interactions with male participants were more efficient (fewer turns, less time).

- **Subjective results**
  - All scores went down from pre-test to post-test.
  - German participants gave systematically lower pre-test scores, but the post-test scores will similar across the groups.
  - The planning domain didn’t have an overall effect. However, German speakers gave the task-based version lower scores for animacy, likeability, and perceived intelligence.

<table>
<thead>
<tr>
<th>Measure</th>
<th>Mean</th>
<th>Median</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drinks served</td>
<td>2.63</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Low ASR turns</td>
<td>6.25</td>
<td>6</td>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>Timeouts</td>
<td>1.05</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>Order requests</td>
<td>7.10</td>
<td>6</td>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td>Response time</td>
<td>14.7</td>
<td>12.9</td>
<td>1.8</td>
<td>52.8</td>
</tr>
<tr>
<td>Time to first drink</td>
<td>51.0</td>
<td>46.1</td>
<td>33.0</td>
<td>110.1</td>
</tr>
<tr>
<td>Total time (s)</td>
<td>111.6</td>
<td>109.0</td>
<td>41.7</td>
<td>214.5</td>
</tr>
<tr>
<td>Total system turns</td>
<td>13.2</td>
<td>13</td>
<td>6</td>
<td>35</td>
</tr>
<tr>
<td>Group orders</td>
<td>0.75</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>2.55</td>
<td>2</td>
<td>1</td>
<td>7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Measure</th>
<th>Task-only (sd)</th>
<th>Full social (sd)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to first drink</td>
<td>55.6 (17.0)</td>
<td>46.3 (15.5)</td>
</tr>
<tr>
<td>Total system turns</td>
<td>11.0 (4.8)</td>
<td>15.5 (5.8)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Category</th>
<th>Pre (sd)</th>
<th>Post (sd)</th>
<th>Change (sd)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthropomorphism</td>
<td>2.74 (0.95)</td>
<td>2.02 (0.64)</td>
<td>-0.73 (0.90)</td>
</tr>
<tr>
<td>German</td>
<td>2.34 (0.78)</td>
<td>1.90 (0.60)</td>
<td>-0.43 (0.86)</td>
</tr>
<tr>
<td>English</td>
<td>3.51 (0.76)</td>
<td>2.24 (0.65)</td>
<td>-1.27 (0.71)</td>
</tr>
<tr>
<td>Animacy</td>
<td>3.04 (0.90)</td>
<td>2.30 (0.70)</td>
<td>-0.74 (0.96)</td>
</tr>
<tr>
<td>German</td>
<td>2.83 (0.97)</td>
<td>2.25 (0.66)</td>
<td>-0.59 (1.01)</td>
</tr>
<tr>
<td>English</td>
<td>3.43 (0.59)</td>
<td>2.42 (0.78)</td>
<td>-1.01 (0.82)</td>
</tr>
<tr>
<td>Likeability</td>
<td>4.43 (1.01)</td>
<td>3.70 (1.17)</td>
<td>-0.74 (1.34)</td>
</tr>
<tr>
<td>German</td>
<td>4.27 (1.05)</td>
<td>4.05 (0.96)</td>
<td>-0.22 (1.01)</td>
</tr>
<tr>
<td>English</td>
<td>4.74 (0.92)</td>
<td>3.04 (1.27)</td>
<td>-1.70 (1.38)</td>
</tr>
<tr>
<td>Perc. Intelligence</td>
<td>3.77 (0.80)</td>
<td>2.87 (0.83)</td>
<td>-0.90 (0.86)</td>
</tr>
<tr>
<td>German</td>
<td>3.52 (0.78)</td>
<td>2.92 (0.90)</td>
<td>-0.60 (0.81)</td>
</tr>
<tr>
<td>English</td>
<td>4.22 (0.65)</td>
<td>2.76 (0.70)</td>
<td>-1.46 (0.64)</td>
</tr>
<tr>
<td>Perc. Safety</td>
<td>4.15 (1.10)</td>
<td>3.97 (1.12)</td>
<td>-0.18 (1.20)</td>
</tr>
<tr>
<td>German</td>
<td>4.12 (1.11)</td>
<td>4.17 (1.08)</td>
<td>+0.05 (0.82)</td>
</tr>
<tr>
<td>English</td>
<td>4.22 (1.12)</td>
<td>3.62 (1.14)</td>
<td>-0.60 (1.65)</td>
</tr>
</tbody>
</table>

(Giuliani et al. 2013)
Ongoing work

- Combining high-level symbolic planning with robot-level motion planning.
- An external motion planner is invoked during generation of the high-level symbolic plan.
- 3D geometric volumes act as a natural intermediate representation for bridging between motion planning and high-level task planning (Gaschler et al. 2013a,b,c).
Conclusions

• Social interaction places additional requirements on the components of a cognitive robotics system: achieving a task goal isn’t always enough.

• The same mechanisms used for general-purpose, symbolic AI planning have been applied successfully to problems in task-based human-robot interaction, as an alternative to more mainstream approaches of interaction management.

• Planning time in the bartender domain is typically quite short and doesn’t negatively impact the system’s reaction time. The choice of planning domain had some influence, however, it was much more subtle than expected. Replanning is frequent.

• Ongoing work: uncertainty reasoning, planning with multiagent knowledge.

• The application area offers a potential testbed for exploring other types of planning problems: planning with preferences, planning under uncertainty, planning with constraints, ...
Thanks to the many researchers who contributed to the design, implementation, and evaluation of the JAMES robot system, especially:

- Mary Ellen Foster (Heriot-Watt University)
- Andre Gaschler (fortiss GmbH)
- Manuel Giuliani (fortiss GmbH)
- Amy Isard (University of Edinburgh)
- Maria Pateraki (FORTH)
- Markos Sigalas (FORTH)
- Richard Tobin (University of Edinburgh)

This research received funding from the European Commission's Seventh Framework Programme (Grant No. 270435).
References


References ... (2)


References ... (3)


For more information about the **JAMES Project**, please visit [http://james-project.eu/](http://james-project.eu/).