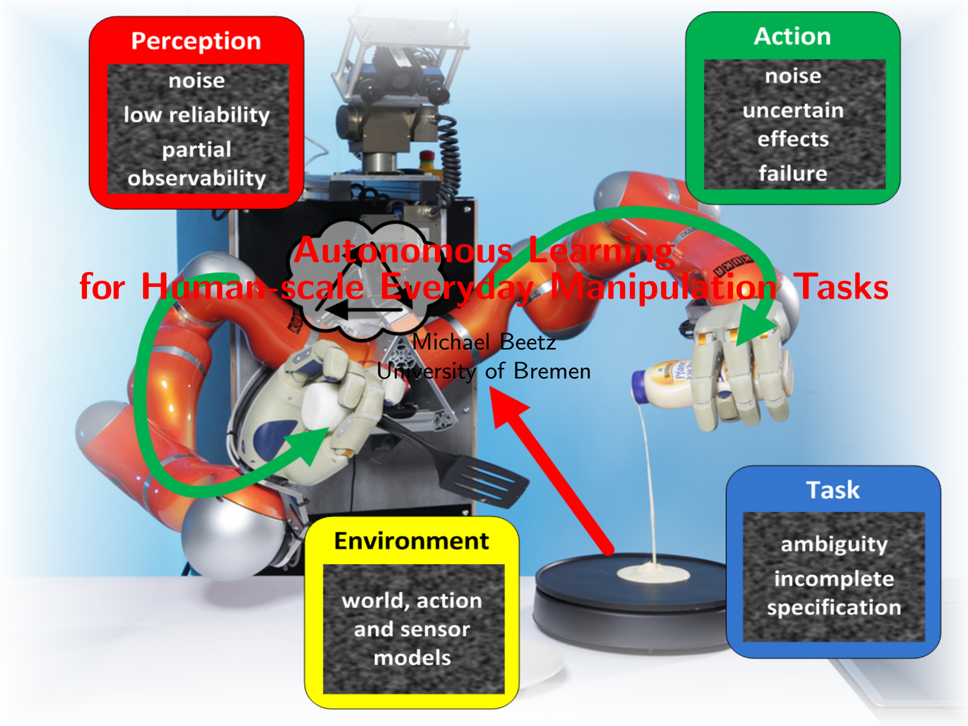
Autonomous Learning for Human-scale Everyday Manipulation Tasks

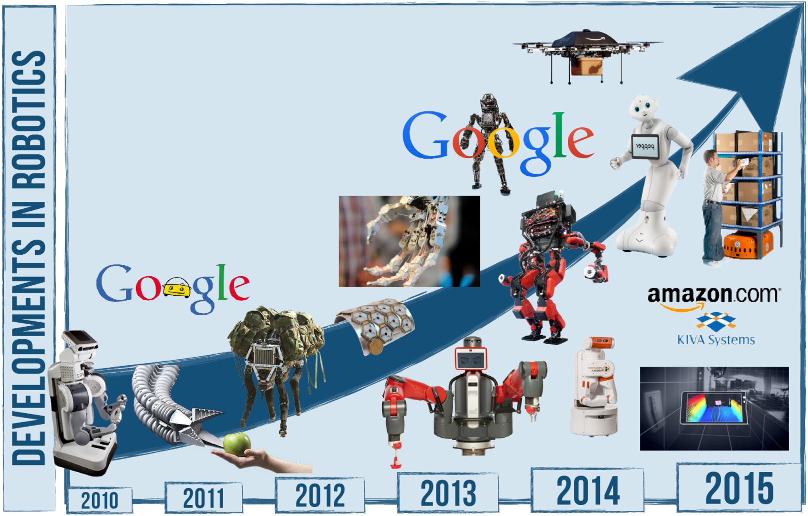
Michael Beetz
University of Bremen



The Research Field:

Autonomous Robotics

The Evolution of (AI-enabled) Robotics



Autonomous Robotics has become a Disruptive Technology

Disruptive technologies =
technologies that will transform life, business, and the global economy

source: Report from McKinsey Global Institute, May 2013

Disruptive Technologies 1-6	Disruptive Technologies 7-12
1. Mobile Internet	7. Next-generation genomics
2. Automation of knowledge work	8. Energy storage
3. The Internet of Things	9. 3D printing
4. Cloud technology	10. Advanced materials
5. Advanced robotics	11. Advanced oil/gas exploration/recovery
6. (Near-)autonomous vehicles	12. Renewable energy

Disruptive Technology Systems

Autonomous Driving

[Google]

Watson

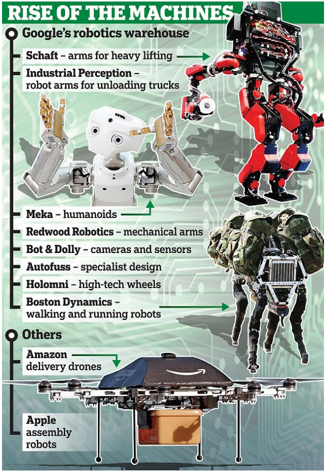
[IBM]

Siri Agent

[Siri/Apple]

- **Google glasses:** knowing everything about what you see (Google goggles)
- **Interactive virtual reality games:** Oculus Rift, Kinect 2, Leap motion sensor, etc
- ...

The Google Disruptive Technology Robot



- **application???**
  - warehouse robot??? picking items on an order list and loading them in packages???
  - delivery robot??? delivering items to people's homes
- **expected capability**
  - capable perception-guided autonomous manipulation
  - longterm autonomy

If you know the solution before understanding the problem you can be sure to be wrong.

Drew McDermott

The formulation of a problem is often more essential than its solution.

Albert Einstein

The Research Problem:

Human-scale Manipulation Tasks

A problem well put is half solved.

John Dewey

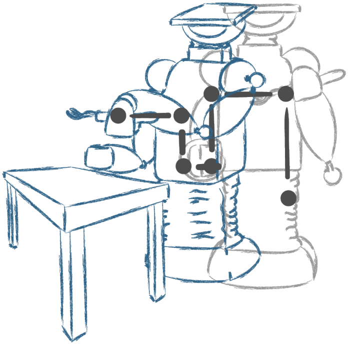
If I had an hour to solve a problem,
I'd spend 55min thinking about the problem
and 5 minutes thinking about solutions.

Albert Einstein

Autonomous Robotic Agents

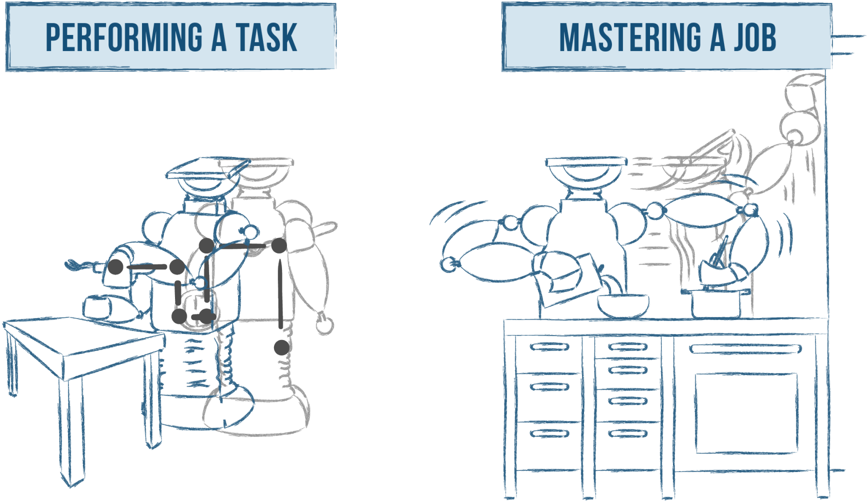
Where we are, where we are going

PERFORMING A TASK



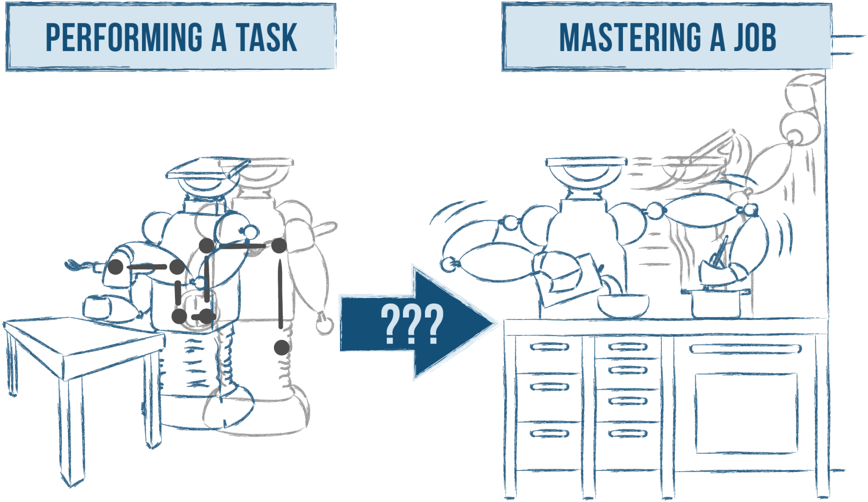
Autonomous Robotic Agents

Where we are, where we are going



Autonomous Robotic Agents

Where we are, where we are going



The Holy Grail: Goal-directed Object Manipulation

evolution of cognitive capabilities:

- representation
- language
- cultural learning

Classical AI Answer

given:

initial state
ingredients
& tools

goal:
have(pancakes)

Classical AI Answer

given:

initial state
ingredients
& tools

goal:
have(pancakes)

compute plan:

pour (pancakemix, bottle, oven)

wait (3min)

flip (pancake)

wait (3min)

put (pancake,plate)

Classical AI Answer

given:

initial state
ingredients
& tools

goal:
have(pancakes)

compute plan:

pour (pancakemix, bottle, oven)

wait (3min)

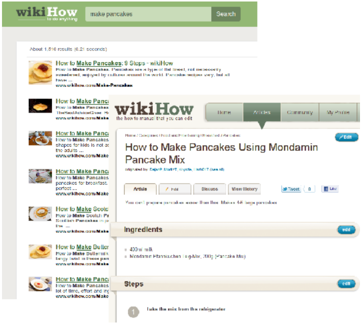
flip (pancake)

wait (3min)

put (pancake,plate)

intelligence lies in the sequencing of actions

Instructions and Actions



tools:

- frying pan
- spatula

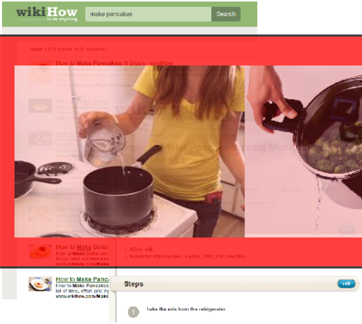
ingredients:

- pancake mix
- milk

Steps:

- Take the mix from the refrigerator.
- Add 400ml of milk; shake the bottle head down for 1 Minute. Let the pancake-mix sit for 2-3 minutes, shake again.
- Pour the mix into the frying pan.
- Wait for 3 minutes.
- Flip the pancake around.
- Wait for 3 minutes.
- Place the pancake onto a plate.

Instructions and Actions



tools:
- frying pan
- spatula

ingredients:
- pancake mix
- milk

Steps:
- Take the mix from the refrigerator.
- Add 400ml of milk, shake the bottle head down for 1 Minute. Let the pancake-mix sit for [illegible] minutes, shake again.
- Pour the mix into the frying pan.
- Wait for 3 minutes.
- Flip the pancake around.
- Wait for 3 minutes.
- Place the pancake onto a plate.

Key Concept: Action Descriptions

Action descriptions represent attributes of actions, objects, etc. that are expected to be important for the skillful execution of actions.

pour stuff from pot

grasp the pot by the handles

hold the pot horizontally

tilt the pot around the axis between the handles

hold the lid while pouring

etc

descriptions can be incomplete, ambiguous, inaccurate, and inconsistent.

Mastering Everyday Manipulation is Knowledge-intensive!

	information in parameterized plans
-	vague instruction (eg, set table, clean up)
=	knowledge required by robotic agents

How much Knowledge Does a Robotic Agent Need?

Knowledge for Mastering Pancake Making

Making a Pancake

A robot pours a ready-made pancake mix onto a preheated pancake maker. Properly performed, the mix is poured onto the center of the pancake maker without spilling where it forms a round shape. The robot lets it cook until the underside of the pancake is golden brown and its edges are dry. Then, the robot carefully pushes a spatula under the pancake, lifts the spatula with the pancake on top, and quickly turns its wrist to put the pancake upside down back onto the pancake maker. The robot waits for the other side of the pancake to cook fully. Finally, it places the pancake using the spatula onto an upturned dinner plate.

What happens if: the robot pours too much pancake mix onto the pancake maker? too little? the robot pours the mix close to the edge of the pancake maker? the robot flips the pancake too soon? too late? the robot pushes only half of the spatula's blade under the pancake? the robot turns its wrist too slow? the robot uses a knife/fork/spoon to flip the pancake? the pancake mix is too thick? too thin? the ingredients of the mix are not homogeneously mixed?

Where Does the Knowledge Come From?

Everyday Activity =

- a complex task that is both common and mundane to the agent performing it;
- one about which an agent has a great deal of knowledge, which comes as a result of the activity being common, and is the primary contributor to its mundane nature; and
- one at which adequate or satisficing performance rather than expert or optimal performance is required.

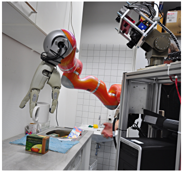
adopted from [Anderson, 1995]

Everyday manipulation is really hard

Picking up an object

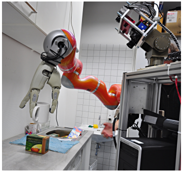
decide on

- where to stand?
- which hand(s) to use?
- how to reach? . . .
- which grasp? where?
- how much force/lift force?
- how to lift? how to hold?



Everyday manipulation is really hard

Picking up an object



decide on

- where to stand?
- which hand(s) to use?
- how to reach? ...
- which grasp? where?
- how much force/lift force?
- how to lift? how to hold?

based on context:

- object, object states, environment, task, ...

Everyday manipulation is really hard

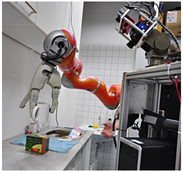
Picking up an object

decide on

- where to stand?
- which hand(s) to use?
- how to reach? ...
- which grasp? where?
- how much force/lift force?
- how to lift? how to hold?

based on context:

- object, object states, environment, task, ...

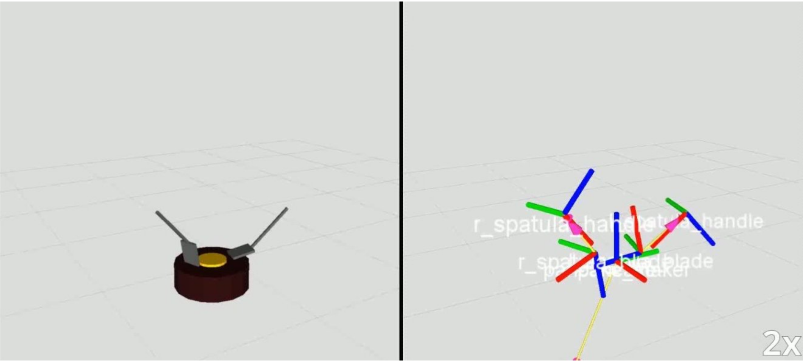


Challenge

- doing the *appropriate* thing
- to the *appropriate* object
- in the *appropriate* way

Manipulation Actions

AI vs control engineering view



AI: symbolic goals, qualitative relations between objects

Control: continuous geometric relations between coordinate frames

Assessment of the Research Problem

Autonomous Learning for Human-scale Manipulation Tasks

- robot control is knowledge intensive
- most knowledge/learning is needed for **how** actions are to be executed
- **hypothesis:** autonomous robot learning of human-scale manipulation tasks is not possible without robots that:
  1. know what they are doing
  2. can read, watch, and play games

- lifelong learning
- learning everything

We investigate 3 aspects

1. the memorization of execution episodes
2. the design of plans
3. the learning of structured joint probability distributions

III

Memories for Autonomous Learning:

Robots that know what they are doing

Memories for Autonomous Learning

III.1 Memories as Query Answering Systems
III.2 Learning Scenarios
III.3 “Big Data” from Manipulation Episodes
III.4 Data Sources for Manipulation Episodes
III.5 KR lite & Data Analytics
III.6 Open-EASE

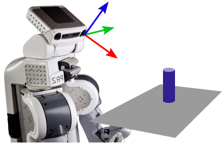
Memorizing Experiences for Autonomous Learning

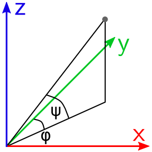
robots that know what they are doing

- can answer queries about
  - what they did,
  - why,
  - what happened,
  - what the effects were,
  - what they saw,
  - what they reasoned,
  - ...

Reasoning about Specific Situations

"Is an object in the assumed field of view?"





Yields explanations for:

- objects not found
- objects possibly occluded

```
?- task_start(log:'CRAMPerceive_uocvmivw', _St),
   owl_individual_of(pr2:pr2_head_mount_kinect_rgb_link, srdl2comp:'Camera'),
   obj_visible_in_camera(log:'VisualPerception_Z9fXhEae_object_0',
                                 pr2:pr2_head_mount_kinect_rgb_link, _St).
true.
```

```
?- task_start(log:'CRAMPerceive_uocvmivw', _St),
   owl_individual_of(Cam, srdl2comp:'Camera'),
   obj_visible_in_camera(log:'VisualPerception_Z9fXhEae_object_0',
                                 Cam, _St).
Cam = pr2:pr2_high_def_frame ;
Cam = pr2:pr2_head_mount_kinect_ir_link ;
Cam = pr2:pr2_head_mount_kinect_rgb_link ;
[...]
false.
```

Objects occluded by robot parts


```
?- task_start(log:'CRAMPerceive_uocvmivw', _St),
   sub_component(pr2:pr2_right_arm, Part),
   obj_blocked_by_in_camera(log:'VisualPerception_Z9fXhEae_object_0',
                            Part,
                            pr2:pr2_head_mount_kinect_rgb_link, _St).
  Part = pr2:pr2_r_wrist_roll_link ;
  Part = pr2:pr2_r_forearm_cam_optical_frame ;
  Part = pr2:pr2_r_gripper_palm_link ;
  [...]
```

Memories for Autonomous Learning

III.1 Memories as Query Answering Systems

III.2 Learning Scenarios

III.3 “Big Data” from Manipulation Episodes

III.4 Data Sources for Manipulation Episodes

III.5 KR lite & Data Analytics

III.6 Open-EASE

Example learning Scenarios

- learning prediction models
- learning perception capabilities
- learning places from which objects can be perceive and reached
- ...

Memories for Autonomous Learning

III.1 Memories as Query Answering Systems

III.2 Learning Scenarios

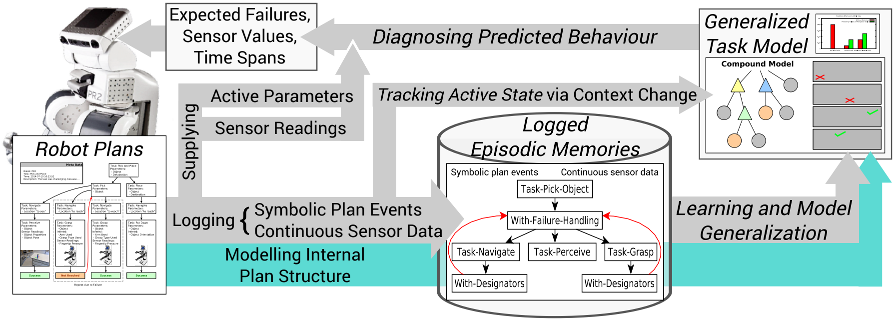
III.3 "Big Data" from Manipulation Episodes

III.4 Data Sources for Manipulation Episodes

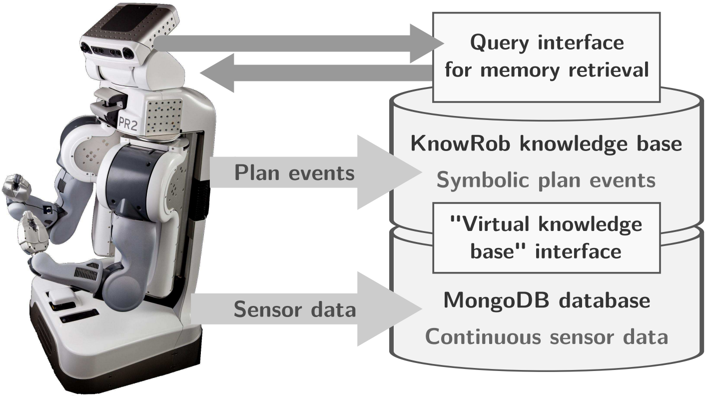
III.5 KR lite & Data Analytics

III.6 Open-EASE

Learning Architecture



Memory System Overview



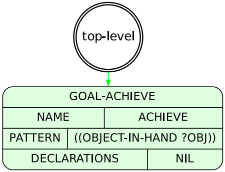
Symbolic Plan Data

Task: **Approach and pick up an object from the kitchen counter**


```
(let* ((loc-desig (a location `((on Cupboard) (name kitchen_island)))))
       (obj-desig (an object `((type container) (at ,loc-desig)))))
  (achieve `(object-in-hand ,obj-desig)))
```

Symbolic Plan Data

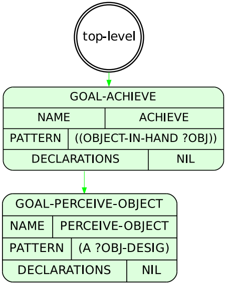
Task: **Approach and pick up an object from the kitchen counter**




```
(let* ((loc-desig (a location `((on Cupboard) (name kitchen_island)))))
       (obj-desig (an object `((type container) (at ,loc-desig)))))
  (achieve `(object-in-hand ,obj-desig)))
```

Symbolic Plan Data

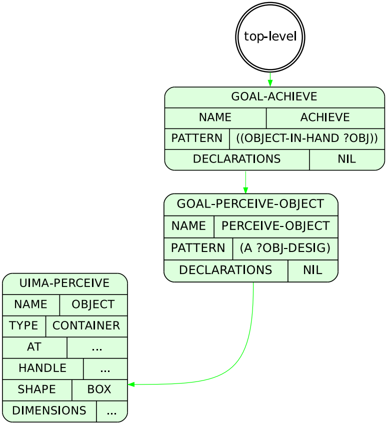
Task: **Approach and pick up an object from the kitchen counter**




```
(let* ((loc-desig (a location `((on Cupboard) (name kitchen_island)))))
       (obj-desig (an object `((type container) (at ,loc-desig)))))
  (achieve `(object-in-hand ,obj-desig)))
```

Symbolic Plan Data

Task: **Approach and pick up an object from the kitchen counter**




```
(let* ((loc-desig (a location `((on Cupboard) (name kitchen_island)))))
       (obj-desig (an object `((type container) (at ,loc-desig)))))
  (achieve `(object-in-hand ,obj-desig)))
```

Symbolic Plan Data

Task: **Approach and pick up an object from the kitchen counter**

Some Object			
TYPE	CONTAINER		
NAME	OBJECT		
POSE	frame-id	"head_mount_kinect_rgb_optical_frame"	
	position	x	-0.118927
		y	0.448207
		z	1.37835
	orientation	x	0
		y	0
		z	0
		w	1

top-level

GOAL-ACHIEVE	
NAME	ACHIEVE
PATTERN	((OBJECT-IN-HAND ?OBJ))
DECLARATIONS	NIL

GOAL-PERCEIVE-OBJECT	
NAME	PERCEIVE-OBJECT
PATTERN	(A ?OBJ-DESIG)
DECLARATIONS	NIL

detected object

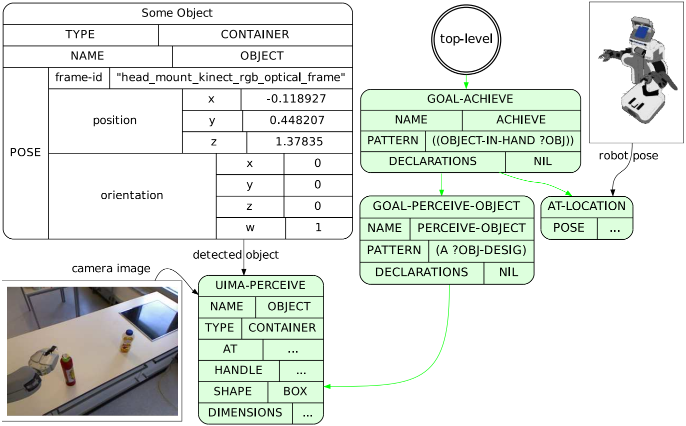
camera image

UIMA-PERCEIVE	
NAME	OBJECT
TYPE	CONTAINER
AT	...
HANDLE	...
SHAPE	BOX
DIMENSIONS	...

```
(let* ((loc-desig (a location `((on Cupboard) (name kitchen_island)))))
       (obj-desig (an object `((type container) (at ,loc-desig)))))
  (achieve `(object-in-hand ,obj-desig)))
```

Symbolic Plan Data

Task: **Approach and pick up an object from the kitchen counter**

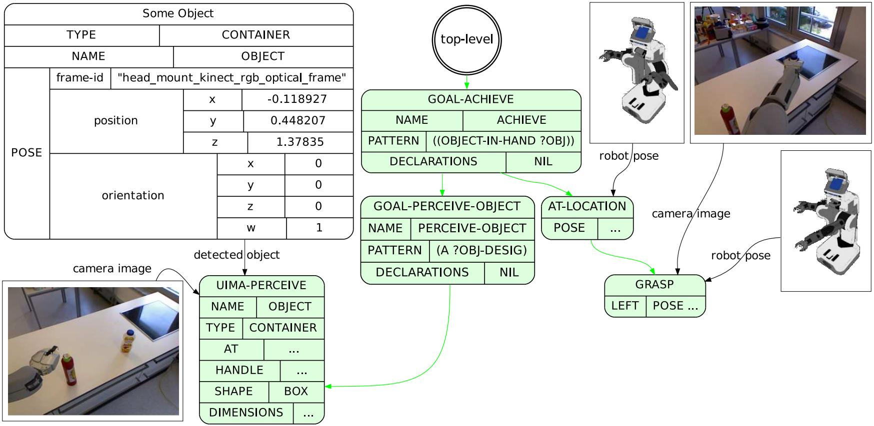



```
(let* ((loc-desig (a location `((on Cupboard) (name kitchen_island)))))
       (obj-desig (an object `((type container) (at ,loc-desig)))))
  (achieve `(object-in-hand ,obj-desig)))
```

Symbolic Plan Data

Task: **Approach and pick up an object from the kitchen counter**

Some Object			
TYPE	CONTAINER		
NAME	OBJECT		
POSE	frame-id	"head_mount_kinect_rgb_optical_frame"	
	position	x	-0.118927
		y	0.448207
		z	1.37835
	orientation	x	0
		y	0
		z	0
		w	1




```
(let* ((loc-desig (a location `((on Cupboard) (name kitchen_island))))
       (obj-desig (an object `((type container) (at ,loc-desig))))
  (achieve `(object-in-hand ,obj-desig)))
```

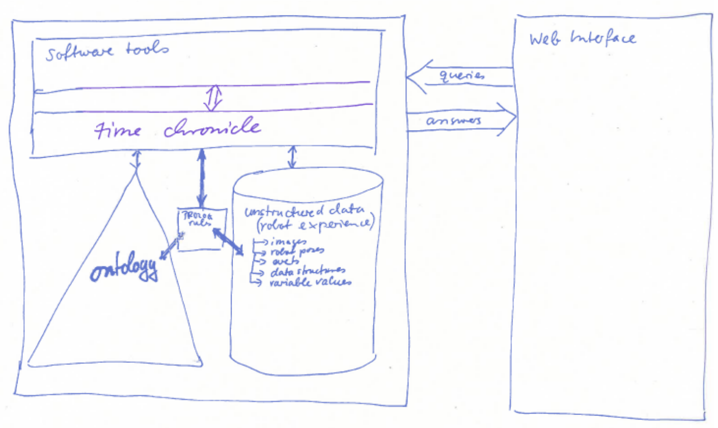
Memories for Autonomous Learning

III.1 Memories as Query Answering Systems
III.2 Learning Scenarios
III.3 “Big Data” from Manipulation Episodes
III.4 Data Sources for Manipulation Episodes
III.5 KR lite & Data Analytics
III.6 Open-EASE

Memories for Autonomous Learning

III.1 Memories as Query Answering Systems
III.2 Learning Scenarios
III.3 “Big Data” from Manipulation Episodes
III.4 Data Sources for Manipulation Episodes
III.5 KR lite & Data Analytics
III.6 Open-EASE

OpenEASE Architecture



Predicates on Experiences

Meta-Predicates (belief state or ground truth)	
holds(*occ*, T_i)	Occasions in the real world
belief_at(*event*, T_i)	Occasions in the belief state
occurs(*event*, T_i)	Events in the belief state
Reasoning about the logged task tree	
task(*Task*)	Tasks on interpretation stack
task_class(*Task*, *Class*)	Class of task
task_goal(*Task*, *Goal*)	Goal of task
task_start(*task*, *T*)	Start time of task
task_end(*Task*, *T*)	End time of task
task_status(*Task*, *Status*)	Status of task (not started, ongoing or finalized)
subtask(*Task*, *Subtask*)	Task is a parent of Subtask
$subtask^+$(*Task*, *Subtask*)	Task is an ancestor of Subtask
returned_value(*Task*, *Result*)	Result of task (success or fail)
failure_task(*Error*, *Class*)	Failure of task
failure_class(*Error*, *Class*)	Class of failures
failure_attribute(*Err*, *Name*, *Val*)	Attribute of failure

Reasoning about events	
loc_change(*Obj*)	Object changed its location
object_perceived(*Obj*)	Object has been perceived
Reasoning about occasions	
loc(*obj*, *Loc*)	Location of an object
object_visible(*Obj*)	Object is visible to the robot
object_placed_at(*Obj*, *loc*)	Object was placed at location
Reasoning about logged poses and designators	
designator_type(*Desig*, *Type*)	Type of designator
designator_props(*Desig*, *Prop*, *Val*)	Property values of designator
obj_pose_by_desig(*Obj*, *Pose*)	Object pose from perceived designator
lookup_transform(*Src*, *Tgt*, *T*, *Tf*)	Logged transform from Src to *Tgt* at time *T*
transform_pose(P_i, *Src*, *Tgt*, *T*, P_o)	Transform P_i from frame Src to frame Tgt at time *T*
visible_in_cam(*Obj*, *Cam*, *T*)	At time *T*, *Obj* was in the field of view of *Cam*
blocked_by_in_cam(*Obj*, *Blk*, *Cam*, *T*)	At time *T*, *Blk* was blocking the view of *Cam* on *Obj*

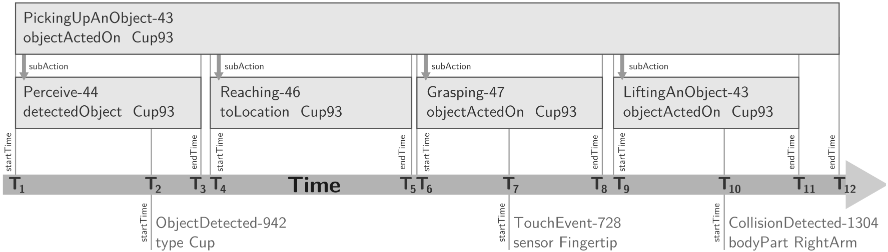
Queries based on Experiences

“Which objects were believed to be on the table?”

“Object occluded by the robot’s arm?”

“What are common failures during pick and place?”

“How probable is success of pick and place after n fails?”

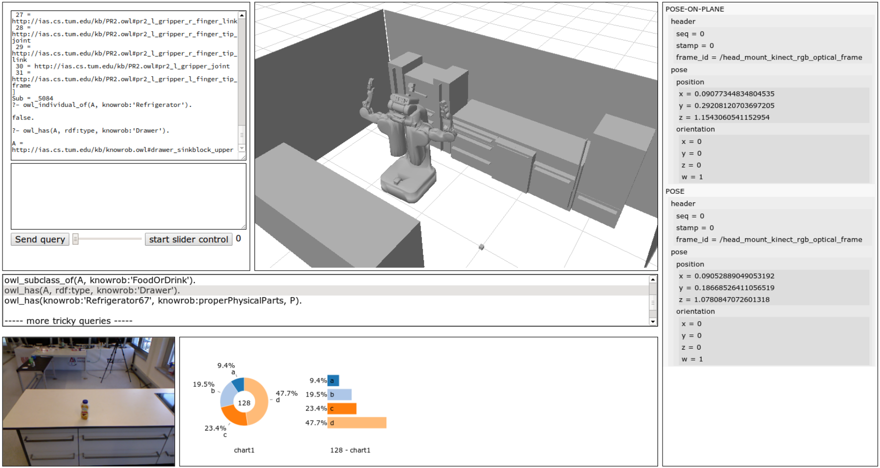


Memories for Autonomous Learning

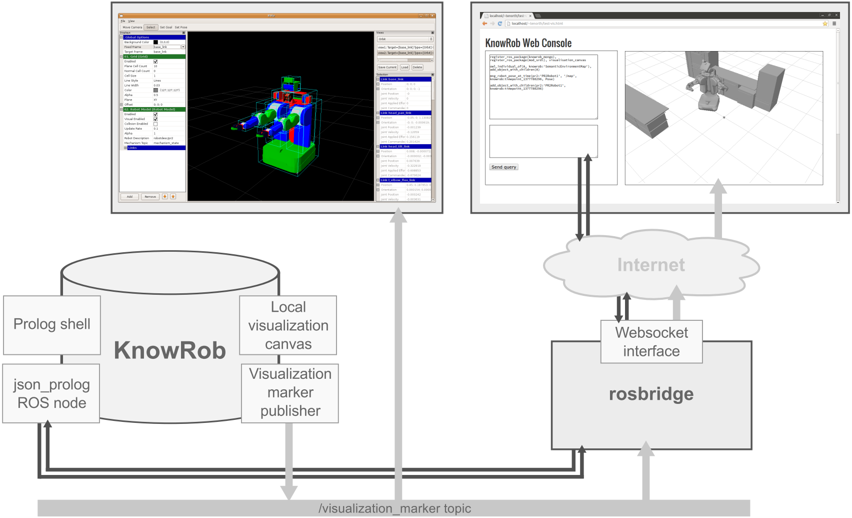
III.1 Memories as Query Answering Systems
III.2 Learning Scenarios
III.3 "Big Data" from Manipulation Episodes
III.4 Data Sources for Manipulation Episodes
III.5 KR lite & Data Analytics
III.6 Open-EASE

A Web interface to KnowRob

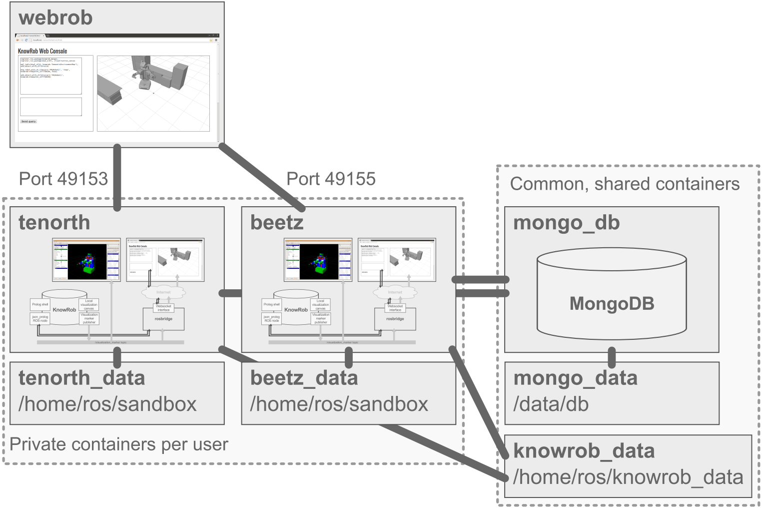
KnowRob Web Console



Implementation using ROS components



Dockerizing KnowRob



Plan Design

Motivation

Focus:

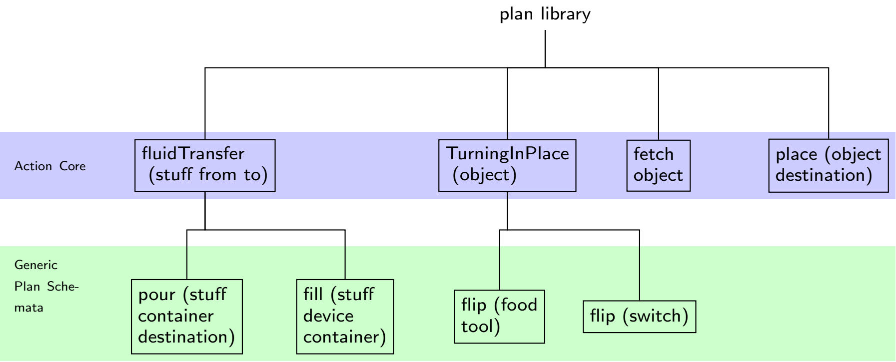
- human-scale everyday manipulation activities
- lifelong learning and adaptation
- "smart" and general plans

Example Plan: Fetch and Place

Plan schema Generalized fetch and place (partial object description)

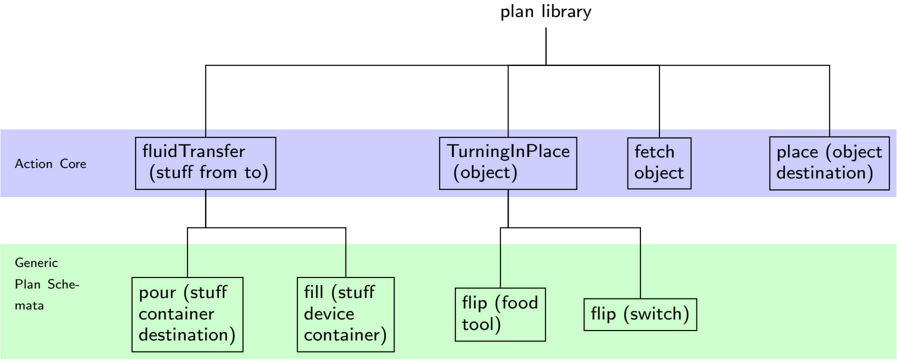
- find an object matching the partial object description
  - go to the place where you believe the object to be
  - look for it; if necessary make it visible
- position yourself in order to grasp the object properly
- ...

An Example Plan Acquisition Episode



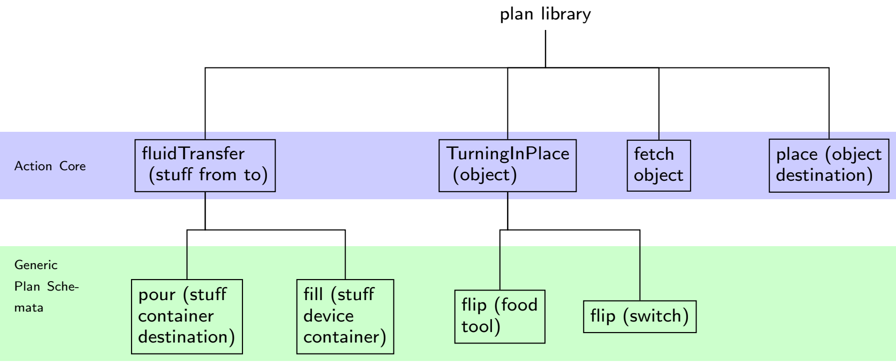
"pour the mix into the frying pan"

An Example Plan Acquisition Episode



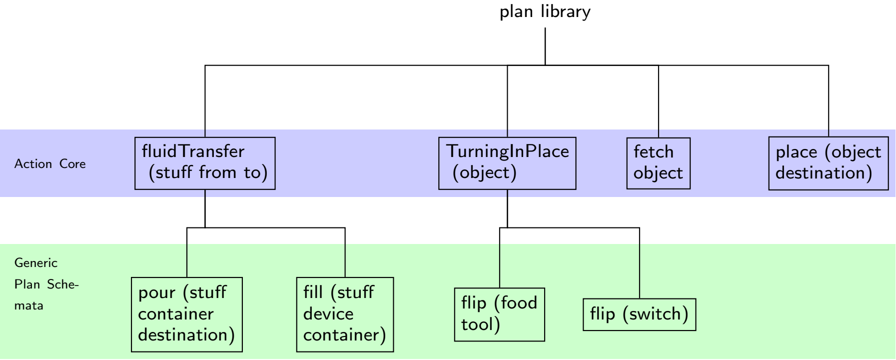
"pour the mix into the frying pan"

An Example Plan Acquisition Episode



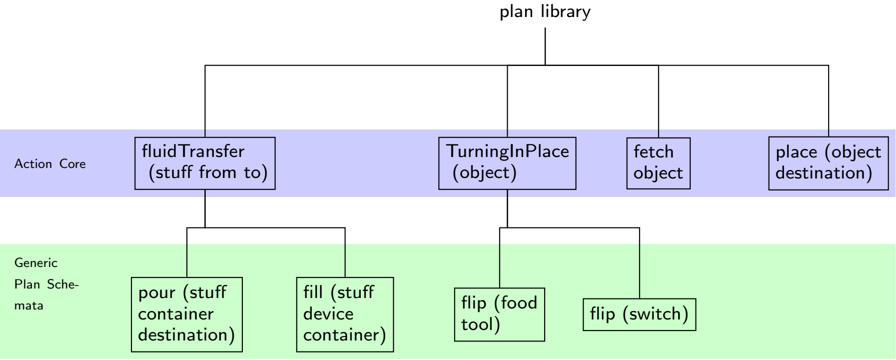
"pour the mix into the frying pan"

An Example Plan Acquisition Episode

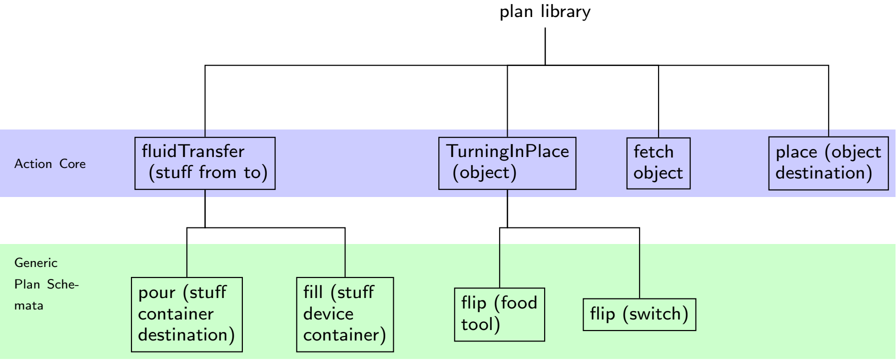


plan acquisition process

An Example Plan Acquisition Episode



An Example Plan Acquisition Episode



Result of the Instruction Interpretation Step

```
plan schema pour( st : (some stuff (type pancake-mix))
                  c : (an object (type container) (contains st))
                  dest : (a location (in frying pan)))
   (with-roles ((stuff st)
                (container c)
                (destination dest))
      (with-subactions ((take-container (an action
                                             (type take)
                                             (object-acted-on container)))
                        (pouring-action (an action
                                             (type pouring)
                                             ...))
                        ...)
         (perform take-container)
         (perform pouring-action)))
```

Step 2: Refinement of Action Descriptions

WP1, WP2, WP5 pour the pancake mix into the frying pan

given:

(an action
    (type pouring)
    (some stuff
        (type pancake-mix))
    (destination
        (a location
            (in *frying pan*))))

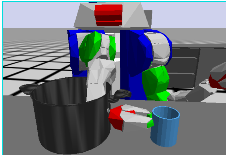
compute:

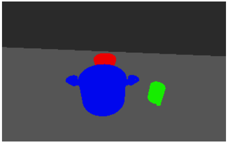
additional attributes that might be necessary to ensure successful execution, such as

- how the container is grasped
- how the container is held
- the pose of the container while the pouring event occurs
- the effect of pouring: amount? shape?
- movement phases and constraints

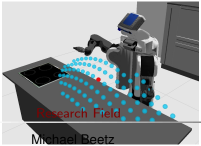
Context-directed Plan Parameterization

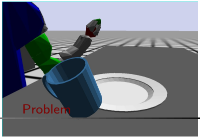
Put the pancake mix away





```
(perform (an action
            (type put-away)
            (object ?obj = (the object
                              (type pancake-mix)))
            (destination ?loc = (a location
                                   (on counter)
                                   (stable ?obj)
                                   (reachable t)
                                   (visible-for James)
                                   (not (hindering (the activity
                                                     (type pancake-making))))))))
```





An Example of Context-directed Plan Parameterization

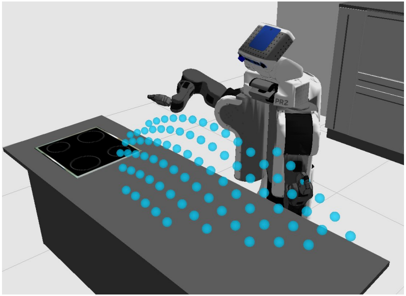
setof ?Pose On(Counter, ?Pose) ?Poses $\wedge$ member(?P, ?Poses)
$\wedge$ Pose(Cup, ?P) $\wedge$ stable(Cup)

An Example of Context-directed Plan Parameterization

setof ?Pose On(Counter, ?Pose) ?Poses $\wedge$ member(?P, ?Poses) $\wedge$ Pose(Cup, ?P) $\wedge$ stable(Cup)

- setof ?Pose On(Counter, ?Pose) ?Poses
- member(?P, ?Poses)
- Pose(Cup, ?P)
- stable(Cup)

Create distribution for sampling poses

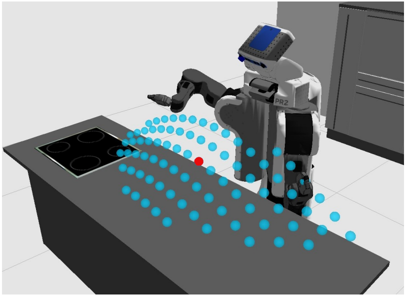


An Example of Context-directed Plan Parameterization

setof ?Pose On(Counter, ?Pose) ?Poses ∧ member(?P, ?Poses)
∧ Pose(Cup, ?P) ∧ stable(Cup)

- setof ?Pose On(Counter, ?Pose) ?Poses
- member(?P, ?Poses)
- Pose(Cup, ?P)
- stable(Cup)

Draw a pose sample

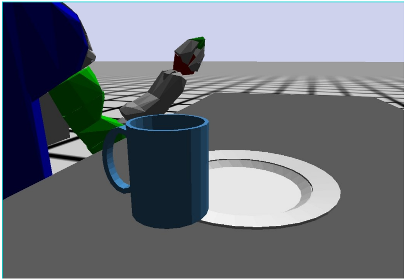


An Example of Context-directed Plan Parameterization

setof ?Pose On(Counter, ?Pose) ?Poses ∧ member(?P, ?Poses)
∧ Pose(Cup, ?P) ∧ stable(Cup)

- setof ?Pose On(Counter, ?Pose) ?Poses
- member(?P, ?Poses)
- Pose(Cup, ?P)
- stable(Cup)

Place the mug

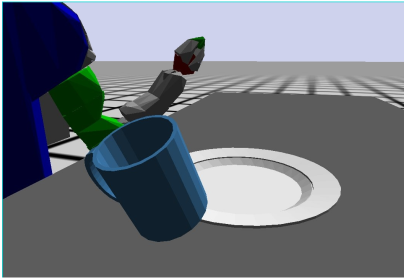


An Example of Context-directed Plan Parameterization

setof ?Pose On(Counter, ?Pose) ?Poses ∧ member(?P, ?Poses)
∧ Pose(Cup, ?P) ∧ stable(Cup)

- setof ?Pose On(Counter, ?Pose) ?Poses
- member(?P, ?Poses)
- Pose(Cup, ?P)
- stable(Cup)

Simulate for 50ms, fail!

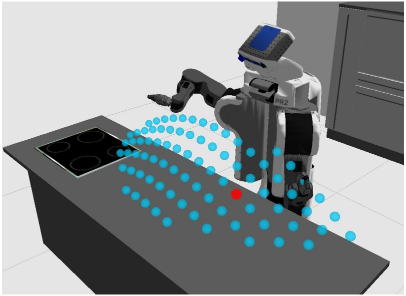


An Example of Context-directed Plan Parameterization

setof ?Pose On(Counter, ?Pose) ?Poses ∧ member(?P, ?Poses)
∧ Pose(Cup, ?P) ∧ stable(Cup)

- setof ?Pose On(Counter, ?Pose) ?Poses
- member(?P, ?Poses)
- Pose(Cup, ?P)
- stable(Cup)

Backtrack, draw another pose sample

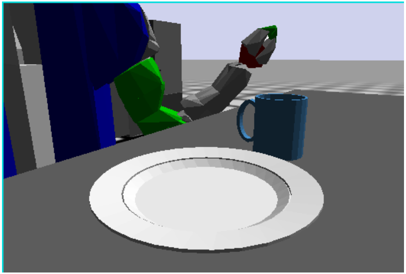


An Example of Context-directed Plan Parameterization

setof ?Pose On(Counter, ?Pose) ?Poses ∧ member(?P, ?Poses)
∧ Pose(Cup, ?P) ∧ stable(Cup)

- setof ?Pose On(Counter, ?Pose) ?Poses
- member(?P, ?Poses)
- Pose(Cup, ?P)
- stable(Cup)

Place the mug

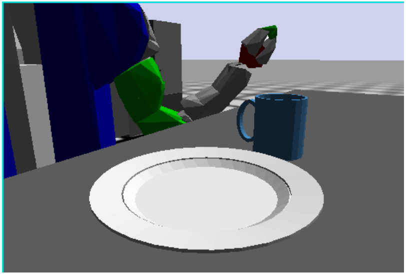


An Example of Context-directed Plan Parameterization

setof ?Pose On(Counter, ?Pose) ?Poses ∧ member(?P, ?Poses) ∧ Pose(Cup, ?P) ∧ stable(Cup)

- setof ?Pose On(Counter, ?Pose) ?Poses
- member(?P, ?Poses)
- Pose(Cup, ?P)
- stable(Cup)

Simulate for 50ms, succeed!



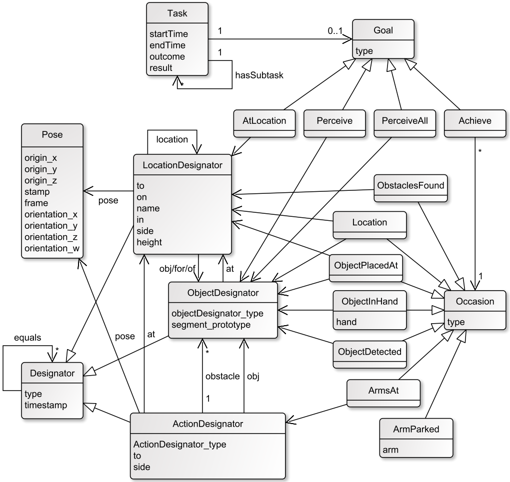
Plan Structure of the Fetch Plan

?*obj* ← Description of an object to grasp

Achieve in-hand (?*obj*)

- ?*locs* ← mostLikelyPlaces(?*obj*)
- For each place ?*loc* in ?*locs* until ?*obj* in hand
  - **At-Location** (a loc (to see ?*obj*) (at ?*loc*))
  - Perceive(?*obj*)
    - Call low-level perception routine
  - Object found?
    - y:
      - Try *n* times at different ?*loc-grasp* (a loc (to grasp ?*obj*))
        - **At-Location** (?*loc-grasp*)
        - Perform (an action (to grasp) (obj-acted-on ?*obj*))
        - Object grasped?
          - y: **Finish**
          - n: Backtrack to new ?*loc-grasp*
    - n: Backtrack to new ?*loc*
      - ∅

Plan Model


Task
startTime
endTime
outcome
result
1
0..1
Goal
type
1
hasSubtask
*
AtLocation
Perceive
PerceiveAll
Achieve
Pose
origin_x
origin_y
origin_z
stamp
frame
orientation_x
orientation_y
orientation_z
orientation_w
location
LocationDesignator
to
on
name
in
side
height
pose
ObstaclesFound
Location
ObjectPlacedAt
obj/for/of
at
ObjectDesignator
objectDesignator_type
segment_prototype
ObjectInHand
hand
Occasion
type
ObjectDetected
equals
Designator
type
timestamp
pose
at
obstacle
obj
1
ArmsAt
ArmParked
arm
ActionDesignator
ActionDesignator_type
to
side

Learning First-order Probabilistic Model

Research Goal of BayCogRob

Answer the Research Question

Can we design plans and provide a computational infrastructure such that plans can autonomously learn a joint probability distribution $P(\mathcal{I}, \mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{E}, \mathcal{R})$ over

- their interpretation $\mathcal{I}$
- the percepts $\mathcal{P}$ they receive and the effects $\mathcal{E}$ they cause
- and the relations $\mathcal{R}$ between $\mathcal{I}$, $\mathcal{P}$, and $\mathcal{E}$

Given $P(\mathcal{I}, \mathcal{P}, \mathcal{A}, \mathcal{T}, \mathcal{E}, \mathcal{R})$

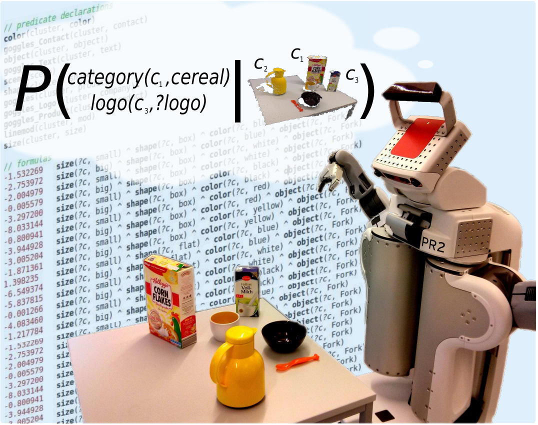
the robot can compute:

- $P(\mathrm{Q} \mid \mathrm{TaskOutcome}(\mathrm{success}, \mathrm{T}) \wedge \mathrm{Context})$
- $P(\mathcal{E} \mid \mathcal{P}, \mathrm{context})$
- ...

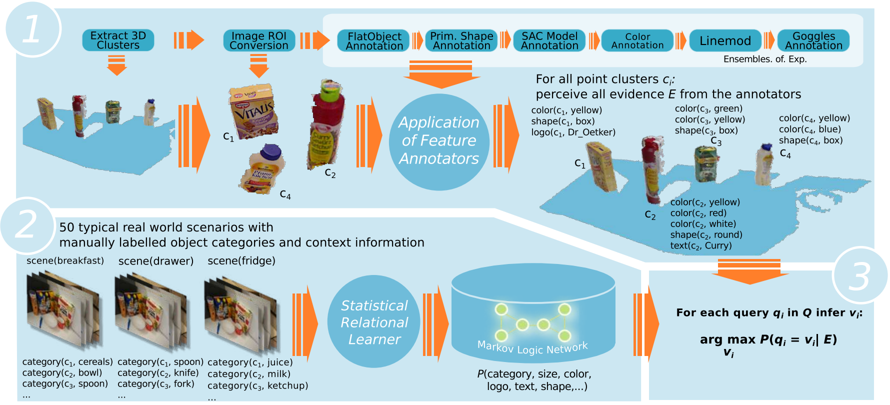
Case Study 1: Learning Models of Perceive Plans

RoboSherlock

[*ICRA'14 (subm.)*]



Case Study 1: Learning Models of Perceive Plans



Case Study 1: Learning Models of Perceive Plans

annotator	annotates if	annotation
Color	always	color(c,col)
Size	always	size(c,s)
Goggles	if Google goggles returns text or logos	logo(c,logo) text(c,text) texture(c,t)
FlatObject	if there are objects that are too flat to be found by the general 3D clustering	shape(c,flat)
PrimShape	always	shape(c,shp)
LineMod	confidence that *c* is one of the objects looked for exceeeds threshold	identity(c,i)
SACmodel	if enough inliers for a model are found	shape(c,sac)
Location	always	scene(c,loc)

Tabelle: Description of the annotators how they work, and what are the resulting annotations.

Case Study 1: Learning Models of Perceive Plans

Prediction/Truth	Bowl	Cereal	Chips	Coffee	Cup	Fork	Juice	Ketchup	Knife	Milk	Mondamin	Oil	Pancake_maker	Pitcher	Plate	Popcorn	Pot	Salt	Spatula	Spoon	Toaster
Bowl	**10**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Cereal	0	**8**	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Chips	0	0	**5**	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Coffee	0	0	0	**9**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Cup	0	0	2	2	**20**	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
Fork	0	0	0	0	0	**1**	0	0	7	0	0	0	0	0	0	0	0	0	0	3	0
Juice	0	1	0	0	0	0	**12**	0	0	1	0	1	0	0	0	1	0	0	0	0	0
Ketchup	0	0	0	1	0	0	0	**7**	0	0	0	0	0	0	0	0	0	0	0	0	0
Knife	0	0	0	0	0	9	0	0	**6**	0	0	0	0	0	0	0	0	0	1	5	0
Milk	0	0	0	0	0	0	3	0	0	**10**	0	0	0	0	0	0	0	0	0	0	0
Mondamin	0	0	0	0	0	0	0	0	0	1	**7**	1	0	0	0	0	0	0	2	0	0
Oil	0	0	0	0	0	0	0	0	0	0	0	**7**	0	0	0	0	0	0	2	0	0
Pancake_maker	0	0	0	0	0	0	0	0	0	0	0	0	**6**	0	0	0	0	0	0	0	0
Pitcher	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	0	0	0	0	0	0	0
Plate	0	0	0	0	0	2	0	0	0	0	0	0	0	0	**23**	0	0	0	3	2	0
Popcorn	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	**5**	0	0	0	0	0
Pot	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**6**	1	1	0	0
Salt	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	**3**	0	0	0
Spatula	0	0	0	0	0	0	0	0	0	0	0	1	0	0	3	0	0	0	**10**	0	0
Spoon	0	0	0	0	0	3	0	0	6	0	0	0	0	0	0	0	0	0	2	**6**	0
Toaster	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	**4**

Case Study 1: Learning Models of Perceive Plans

Google goggles

Prediction/Truth	Bowl	Cereal	Chips	Coffee	Cup	Fork	Juice	Ketchup	Knife	Milk	Mondamin	Oil	Pancake_maker	Pitcher	Plate	Popcorn	Pot	Salt	Spatula	Spoon	Toaster
Bowl	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Cereal	0	**8**	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Chips	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Coffee	0	0	0	**2**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Cup	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Fork	10	0	4	9	20	**15**	0	1	19	7	4	8	6	3	28	5	6	4	16	16	4
Juice	0	2	2	0	0	0	**14**	0	0	3	1	2	0	0	0	1	0	0	3	0	0
Ketchup	0	0	1	0	0	0	0	**5**	0	0	0	0	0	0	0	0	0	0	0	0	0
Knife	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0
Milk	0	0	0	0	0	0	0	0	0	**3**	0	0	0	0	0	0	0	0	0	0	0
Mondamin	0	0	0	0	0	0	1	1	0	1	**1**	0	0	0	0	0	0	0	0	0	0
Oil	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0
Pancake_maker	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0
Pitcher	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0
Plate	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0
Popcorn	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
Pot	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0
Salt	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
Spatula	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	**3**	0	0
Spoon	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
Toaster	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**

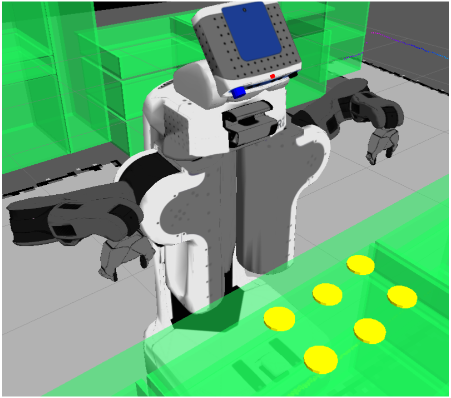
shape annotators

Prediction/Truth	Bowl	Cereal	Chips	Coffee	Cup	Fork	Juice	Ketchup	Knife	Milk	Mondamin	Oil	Pancake_maker	Pitcher	Plate	Popcorn	Pot	Salt	Spatula	Spoon	Toaster
Bowl	**10**	0	0	3	6	0	0	4	0	2	2	4	0	3	4	0	5	3	13	0	0
Cereal	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Chips	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Coffee	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Cup	0	0	7	0	**14**	0	0	3	0	0	5	7	0	0	0	0	0	0	1	0	0
Fork	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Juice	0	10	0	9	0	0	**17**	0	0	13	0	0	6	0	0	6	1	1	1	0	4
Ketchup	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0
Knife	0	0	0	0	0	11	0	0	**12**	0	0	0	0	0	13	0	0	0	2	9	0
Milk	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0
Mondamin	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0
Oil	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0
Pancake_maker	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0
Pitcher	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0
Plate	0	0	0	0	0	4	0	0	7	0	0	0	0	0	**11**	0	0	0	5	7	0
Popcorn	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0
Pot	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0
Salt	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0
Spatula	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0
Spoon	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
Toaster	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**

- high categorization accuracy
- exploiting background knowledge
- exploiting co-occurrence of objects in scenes
- additional kinds of inference tasks

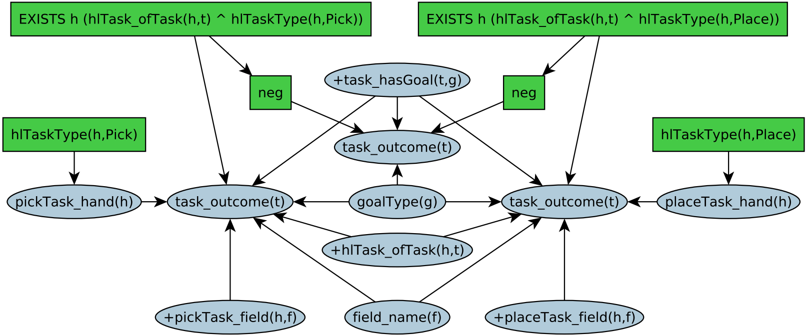
Case Study 2: Learning Models for Toy Pick-and-Place

TUM-James performing a pick and place task:



Case Study 2: Learning Models for Toy Pick-and-Place

Bayesian logic network trained on execution trace data (manually defined structure):



Case Study 2: Learning Models for Toy Pick-and-Place

Query (parametrisation): Which manipulator to use, so that success is more likely for picking up an object from the front-left position?

$$
\begin{aligned}
\mathrm{P}(&\mathrm{pickTask\_hand(H)} \mid \mathrm{task\_hasGoal(T,G)} = \mathrm{True} \;\wedge \\
&\mathrm{task\_outcome(T)} = \mathrm{SUCCEEDED} \;\wedge \\
&\mathrm{hlTask\_ofTask(H,T)} = \mathrm{True} \;\wedge \\
&\mathrm{hlTaskType(H)} = \mathrm{Pick} \;\wedge\; \mathrm{pickTask\_field(H,F)} = \mathrm{True} \;\wedge \\
&\mathrm{field\_name(F)} = \mathrm{FrontLeft})
\end{aligned}
$$

$$\approx \langle\ \mathrm{LEFT: 0.58,\ RIGHT: 0.42}\ \rangle$$

Case Study 2: Learning Models for Toy Pick-and-Place

Query (prediction): What is the probability of being able to successfully place an object at the back-middle position with the right manipulator?

$$
\begin{aligned}
P(&\mathrm{task\_outcome}(T) \mid \mathrm{task\_hasGoal}(T, G) = \mathrm{True} \wedge \\
&\mathrm{hlTask\_ofTask}(H, T) = \mathrm{True} \wedge \\
&\mathrm{hlTaskType}(H) = \mathrm{Place} \wedge \\
&\mathrm{placeTask\_field}(H, F) = \mathrm{True} \wedge \\
&\mathrm{pickTask\_hand}(H) = \mathrm{Right} \wedge \\
&\mathrm{field\_name}(F) = \mathrm{BackMiddle})
\end{aligned}
$$

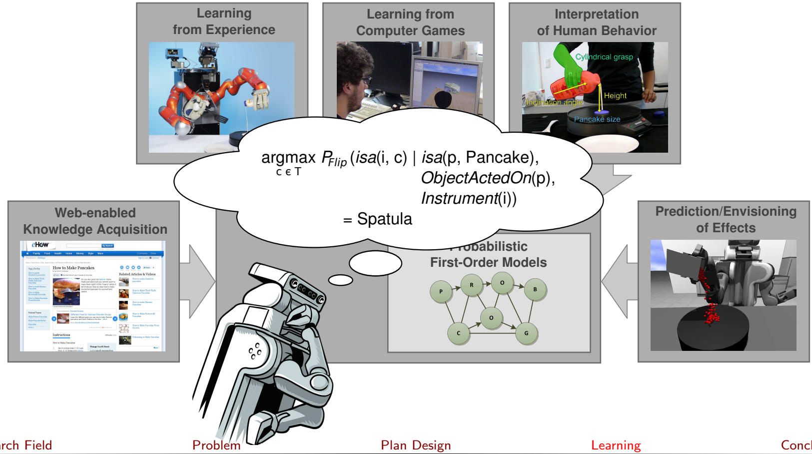
$$\approx \langle\ \mathrm{SUCCEEDED{:}}\ 0.80,\ \mathrm{FAILED{:}}\ 0.20\ \rangle$$

Case Study 3: PRACS — Probabilistic Action Cores

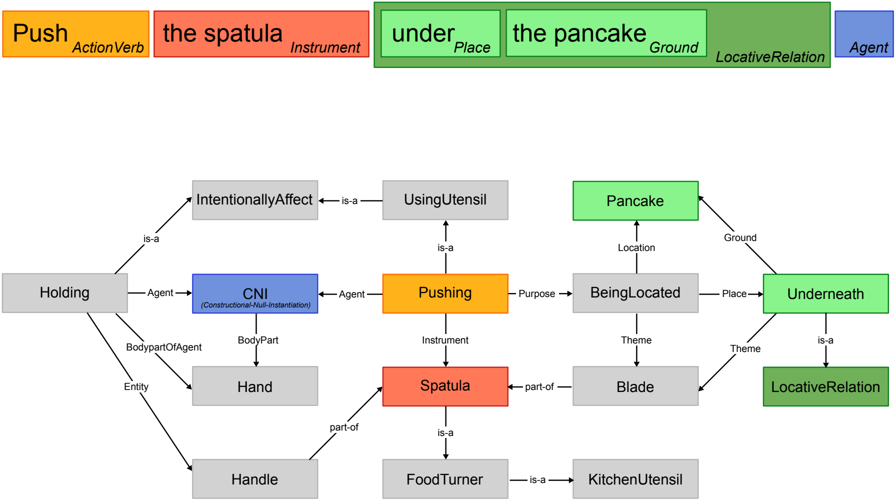
"Flip the pancake!"

[*IROS'12*]

Probabilistic reasoning for disambiguation and filling information gaps



Case Study 3: PRACS — Probabilistic Action Cores



Conclusions

- Bayesian Cognitive Robotics =
  plans that learn probabilistic models of themselves
  - BAYCOGROB so far: understanding by building
  - tremendous potential (demonstration examples with substantial impact)
  - tip of the iceberg

Outlook

- framework for Bayesian Cognitive Robotics
- longterm fetch and place (under realistic conditions)
- learning algorithms that can exploit problem structure
- incremental learning
- query and plan specialization