EU FP7 CogX
ICT-215181
May 1 2008 (52months)

# DR 7.2:
# Analysis of a robot that acts under partial information and uncertainty

⟨cogx@cs.bham.ac.uk⟩

| | |
|---|---|
| *Due date of deliverable:* | Mar 31 2011 |
| *Actual submission date:* | Mar 31 2011 |
| *Lead partner:* | BHAM |
| *Revision:* | draft |
| *Dissemination level:* | PU |

This deliverable reports on insights gathered from evaluation of robots developed in period two for the key scenarios in CogX: Dora, George, and Dexter.

# 1 Introduction

This document is the second in the series of deliverables reporting on evaluation results and insights gathered from studies of the integrated systems developed in CogX. We continue to follow the evaluation and analysis schemes outlined in deliverable D.7.1. Evaluation and assessment of individual components and smaller subsystems are reported in the respective deliverables in the other work packages, while in this document we focus on the evaluation of the integrated systems as a whole.

Our general roadmap in system evaluation is based on an interweaved software release and experiment strategy. Here, we report the relevant building blocks and implementations that led to the major release 2 of the robots Dora and George. The release the evaluation is based on is functionally equivalent to the systems demonstrated at the review in 2010. The focus of this year's integration was to deal with *uncertainty* on a system level. Accordingly have most of the representations and reasoning abilities in the systems been extended or substituted to account for uncertainty in perception and knowledge. In our aim to deal with uncertainty and incomplete knowledge in a theoretically sound way, discrete and deterministic representations have made room for probabilistic (Bayesian) ones. In extension to the last deliverable D.7.1, this time we also report integration and evaluation efforts carried out in the Dexter scenario, where further steps to integrate components from several partners have been made.

We continued to employ CAST [5] as the unifying integration framework, build around the ideas of interacting working memories. CAST is the basis for building systems in all three scenarios and also ensures continuous convergence of the systems. Indeed, Dora and George systems are based on the very same software release, only the orchestration of components and the evaluation setting is different. Also, the hardware platform employed in both scenarios are the same. In order to help the evaluation also from an architectural point of view, the consortium has invested efforts to instrument CAST and the developed system to make them accessible to qualitative and quantitative analyses. As part of these efforts, a generic architecture description language and suitable software metrics have been proposed that support the assessment of particular system architectures [14, in annex A.3]. We have show-cased the ability to generate a model of a running system and analyse it by means of those proposed metrics for a reduced version of the Dora system. In the following, the demonstration scenarios and their evaluation will be discussed in separate sections.

# 2 Dora

## 2.1 Scenario

While in Dora release 1 we emphasised self-extension and exploration, Dora release 2 is focused on the exploitation of and reasoning with uncertain information and uncertain sensing to achieve tasks. While in general Dora is designed and orchestrated to serve a number of different tasks, for the evaluation we chose an object search task. Choosing one single well-defined task makes it possible to run a number of experiments in real world repeatedly so that empirically significant evaluation becomes possible. However, we did not tailor the system to the object search task particularly. Instead, task-independent representations, and reasoning and planning techniques

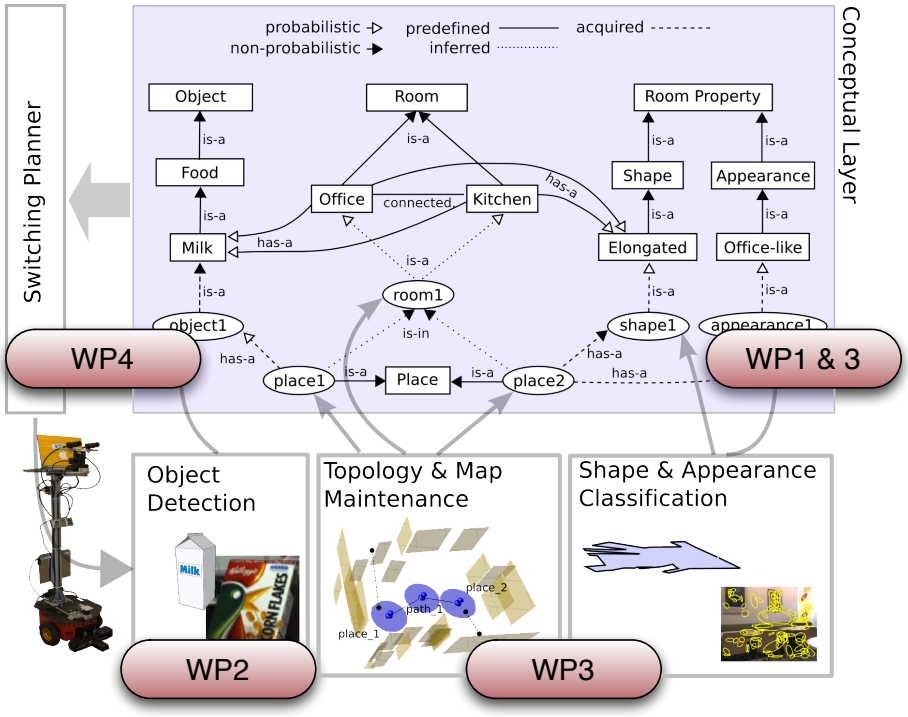



Figure 1: Major processes and conceptual representation employed in Dora 2.

are employed in this year's system. The scenario is particularly chosen to showcase and assess our combination of structural and probabilistic knowledge, the conceptual spatial representation and its reasoning abilities, and the switching planner.

## 2.2 Description of Dora 2

Dora release 2 is a successor of version 1, described in the deliverable D.7.1. Hence, in the following we will describe the modifications and extensions which are mostly devoted to deal with uncertainty in a principled way.

Fig. 1 depicts the most relevant processes and conceptual representation in Dora 2. It shall be noted that this figure does by far not illustrate all the processes but focuses very much on the loop of maintaining a spatial probabilistic, relational representation and planning with it to achieve a given task.

In the following, the relevant building blocks and their role within Dora will be detailed.

**Dialogue management and goal generation:** Dora features the dialogue subarchitecture, responsible for the interaction with human users. In

the current system of Dora, dialogue.sa only is employed to receive tasks from the human. However, it already does this in the very same way – using the same implementation – we employ in the George scenario for the interactive learning. Whenever a human expresses the intention to assign the robot a task, e.g. to find some cornflakes, dialogue.sa's linguistic and formal analysis forms a belief about the human's intention and stores it as an epistemic structure in the working memory. The goal generation and management system, being part of the planning subarchitecture, picks up this intention and by these means adopts the human's intention as a task for the robot. As such, intentions expressed by humans complement the set of goals that were already present in Dora 1 as a result of goal generation from knowledge gaps.

**Topology and map maintenance:** For Dora 2 we kept the place layer, representing the topology of the acquired map of Dora, mostly unmodified from the previous version. Non-monotonic reasoning about the decomposition into rooms and incremental, also active, extension of this representation are still in place and allow to interleave task-driven behaviour with curiosity-driven extension [4, in annex A.2].

**Shape and appearance categorisation:** This entirely new subsystem has been added to provide the system with evidence on the category of rooms. We integrated the approach to categorisation from [15]. It extracts simple geometrical features from laser scans and uses Composed Receptive Field Histograms from acquired camera images as visual features. These features are evaluated on the basis of Support Vector Machine (SVM) models representing specialised concepts of Room Property (e.g. elongated, office-like etc.). Accumulated confidence gained from the SVM models for all views obtained at a topological place are normalised to gain probabilities.

**Conceptual representation and Bayesian reasoning:** In our aim to develop a unifying representation that can accomodate both, instance and conceptual knowledge, we integrated a novel conceptual layer composed of relations between concepts and instances. This model is one of the two major contribution discussed in [2, in annex A.1]. An excerpt of this model in a particular situation is depicted in Fig. 1. Implemented on top of the qualitative spatial framework proposed in [15] and inherent part of Dora's architecture from the very beginning, this layer facilitates inference that accounts for general high-level taxonomic conceptual knowledge, and the outputs of sensing. The former corresponds to a rich probabilistic ontology that specifies categories of room (kitchen, office, etc), how rooms are composed of topological places, and the likelihood that certain types of rooms will contain

particular objects. The outputs of sensing include the results of shape and appearance categorisation mentioned above. We distinguish three primary sources of knowledge: (1) predefined conceptual knowledge, e.g. the likelihood that cornflakes occur in kitchens; (2) knowledge acquired directly according to sensing, e.g. the likelihood of being located adjacent cornflakes given the result of an object detection; and (3) inferred knowledge, e.g. that the room is likely to be a kitchen, because you are likely to have observed cornflakes in it.

For inference, we instantiate the entire conceptual layer including concepts (depicted as rectangles), instances (ellipses) and relation (directed or undirected edges) as a probabilistic graphical model, a *chain graph* representation [9]. Chain graphs provide a natural generalisation of directed (Bayesian Networks) and undirected (Markov Random Fields) graphical models. As such, they allow us to model both "directed" causal as well as "undirected" symmetric or associative relationships, including circular dependencies. In Dora, this allows to present conditional probabilities, such as the "has-a" relation, but also undirected relations. The "connected" relation illustrated in Fig. 1 is one example of the latter. In the chain graph, belief updates are event-driven. Whenever a significant change is detected, either be creating a new instance or removing one or by a change of the probability of a relation, the change is propagated throughout the chain graph model. The underlying inference is approximate to be fast. We employ the fast Loopy Belief Propagation [10] algorithm to realise this inference.

As can be seen from Fig. 1 some of the relations are quantified while others are still deterministic. The probabilistic relations have to be quantified appropriately to draw inferences in the chain graph model. For probabilistic relations between instances and concepts, such as the 'is-a'-relation linking shape properties to concepts like 'Elongated' are derived from the sensing processes. Probabilities quantifying relations between concepts, e.g. 'has-a' between 'Milk' and 'Kitchen' in the example shown, are pre-acquired offline. This particular relation, modeling the cooccurrence of objects of a certain type in specific types of rooms, represents a sort of common-sense knowledge which should be valid in general and shall be exploited when the robot is engaged in a task to search specific objects. Hence, this "has-a" relation linking rooms and objects was bootstrapped using a part of the *Open Mind Indoor Common Sense* database[1]. This contains a list of associations, gathered by asking humans, between common everyday objects and their typical locations, providing us with a set of 5,800 deterministic relations between locations and objects. In order to quantify these relations we then used the object-location pairs to generate '*obj* in

[1] `http://openmind.hri-us.com/`

the *location*' queries to an online image search engine. The number of hits returned was then correlated with the total number of hits for querying '*location*' images, yielding a co-occurrence frequency distribution of approximately 1.5 million object-location pairs. These have been normalised to yield conditional probabilities quantifying the respective relation. The 'has-a' relations linking subtypes of 'Room' and 'Room Property' have been estimated offline from acquired training sets comprising data recorded in a number of different rooms of the respective categories.

**Switching planner:** To generate flexible goal-oriented behaviour the Dora system (and George as well) employs a domain-independent planner. The goal is defined by the goal management system and the current belief state for the planner is given through the probabilistic relational model of the conceptual layer[2]. It shall be noted that this belief state is continuously updated as a consequence of the event-driven inference mechanisms and therefore can trigger replanning in the continual planning approach we integrate in our system.

In order to act intelligently also in the presence of uncertainty in knowledge (e.g., of finding milk in kitchens) and sensing (e.g., failing to see the milk even if it's there), contingent planning techniques are desirable to account for potential failures and to consider alternative solutions in a principled way. However, it turns out that the state spaces we are dealing with in the Dora system are at the order of $\sim 10^{27}$ states, rendering planning on the basis of POMDPs with state of the art techniques an intractable problem given the time constraints of the scenario [2, in annex A.1]. Therefore, we integrated the *switching planner* [1] developed recently in the CogX project. Like the planner in Dora 1 it is a planning system that operates according to the continual planning paradigm. The system *switches*, in the sense that the underlying planning procedure changes depending on our robot's subjective degrees of belief, and progress in plan execution. When the underlying planner is a deterministic sequential planner, i.e., a *classical* planner, we say planning is in a *sequential* session, and otherwise it is in a *contingent* session. Finally, planning is continual in the usual sense that, whatever the session, plans are adapted and rebuilt online in reaction to changes to the planning model (e.g. when objectives are modified, or when our robot's path is obstructed by a door being closed). By autonomously mixing these two types of sessions our robot is able to

[2]In fact, the belief state is still derived from two sources of information: the belief models in the binding subarchitecture encoding the current instance knowledge of the system obtained from sensing and interaction with the user, and the ontology capturing the conceptual default knowledge of the system. However, this distinction is entirely implementation-specific and therefore ignored in the discussion.

be robust and responsive to changes in its environment *and* make appropriate decisions in the face of uncertainty. As such, the switching planner is employed for combined task and observation planning. Predefined observation models enable it take the possibilities of failures in sensing into account. Further details about the switching planner and its role in Dora 2 can be found in the annex A.1.

**View point generation:** While in Dora 1, the actual search for object was very limited and implemented as a monolithic action in the system, Dora 2 featured a so-called *view cone generation* algorithm that makes the actual object search more efficient by constraining the space to search through the exploitation of topological spatial relation [16] developed as part of work package 3 on CogX. In Dora 2 we restricted ourselves to on-ness relations [17] that help to find objects located on tables more efficiently if a table is known to be in a room. View cone generation in this notion describes the process of determining a minimal set of optimal poses and camera orientations maximising the likelihood of finding the object, taking into account default knowledge about the object's common location and relations. The resulting set of this view point generation is available to the planning process to initiation sense action (object detection) in those generated view cones, governed by their estimated likelihood of containing a respective object.

**Object detection:** In contrast to Dora 1 where an off-the-shelf object detection algorithm had been employed, Dora 2 featured the BLORT vision system [11], developed in the CogX project. In the current system we use it to reliably detect pre-trained objects. Object detection is an active process, explicitly planned for by the switching planner.

## 2.3 Evaluation and Analysis Approach

The evaluation carried out with Dora is fully done in real world with the integrated release 2. For the assessment and evaluation we ran the system at two sites (office environment at KTH and a home environment in BHAM, see Fig. 3 and 4, respectively) with a similar setting and the same task. The evaluation approach is two-fold: First we carry out a detailed analysis of the course of actions to understand and justify the robot's behaviour. Second, a quantitative analysis comparing a lesioned system, that does not have access to the evidence obtained from room properties and therefore cannot infer the category of rooms, with the full system is presented.

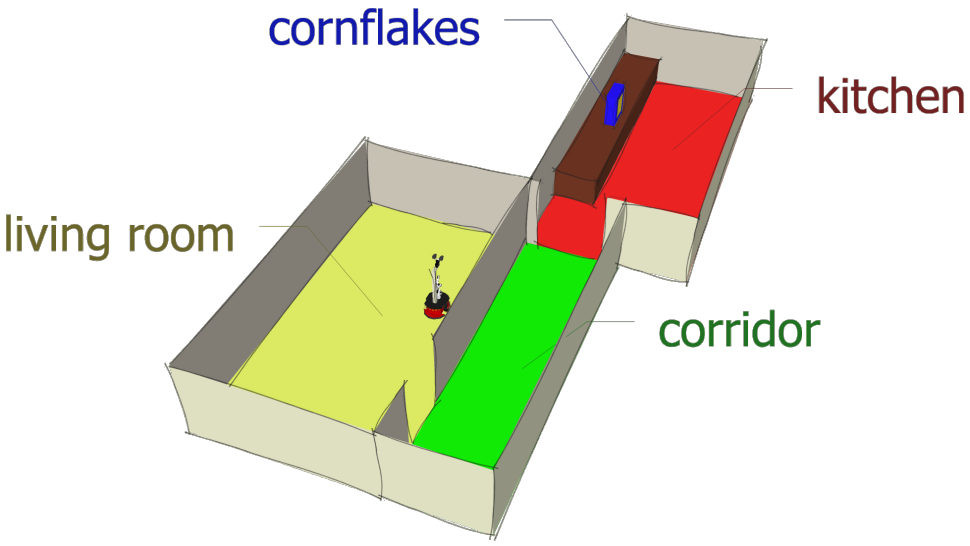



Figure 2: A sketch of the exemplary setup: The robot starts in the living room to search an object that actually is to be found in the kitchen.

### 2.3.1 Analysis of an exemplary course of action

A sketch of the object search setting as it was in BHAM for the evaluation is depicted in Fig. 2. In this setting the object ('cornflakes') that the robot is asked to find is located in the kitchen, while the robot starts in the living room. The following course of actions is also shown in the video to be found in the annex A.6.1 of this report. The following description complements the results presented in [2, in annex A.1]).

1. For this case study we assume that the topological structure of the map remains unchanged throughout the run, i.e. all places are already present as displayed in Fig. 4. Dora starts by autonomously moving (e.g. as part of an exploration strategy) from place 0 to place 2 and recognises room properties for visual appearance and geometric shape for these three places, shown as text attached to the nodes in Fig. 4.

2. The acquisition of these properties triggers inference in the chain graph model leading to an update of the beliefs about the category of the room these places are in. The new evidence is therefore propagated and yields new probabilities for the 'is-a' relation connecting the instance of this room to the subtypes of the 'Room' concept (see Fig 1). The situation resulting from this update is exactly what is shown in Fig. 4 with the pie charts representing the probability distribution over these particular 'is-a' relations. It can be seen, that Dora inferred that the current room is unlikely to be a kitchen, and quite likely a living room.

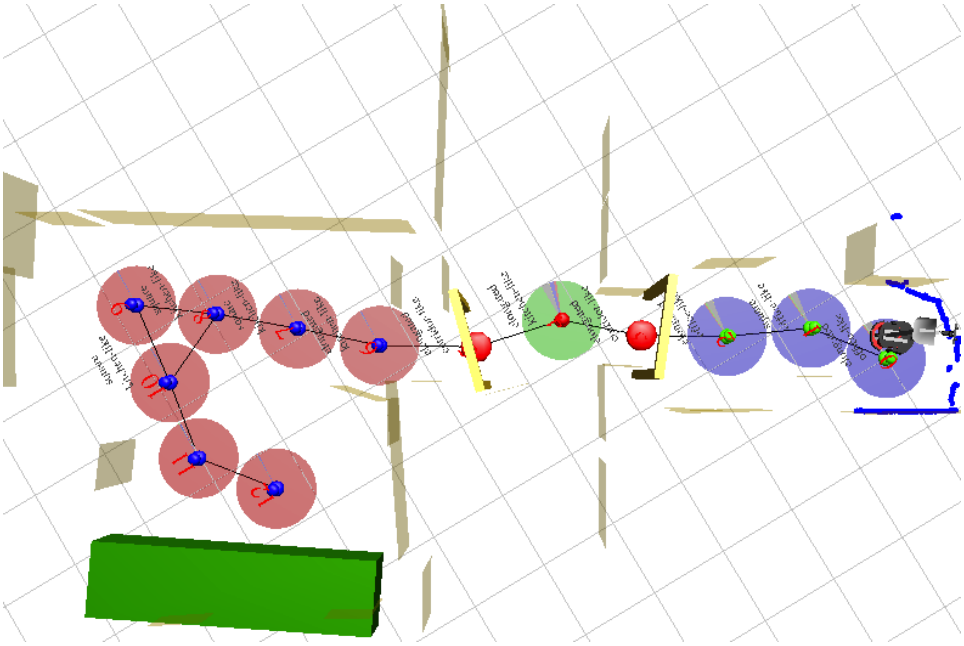


Figure 3: A screenshot of the map acquired in the office setting at KTH.

3. A human says to the robot "Dora, find me some cornflakes". This prompt is processed by the dialogue subarchitecture resulting in an intention attributed to the human representing this request. The intention is then picked up by a goal generation component [3] which translates it into a high priority planning goal. It is assigned a high priority to ensure the robot will instantaneously attend to it.

4. As soon as this new goal is then activated the switching planner generates a sequential plan by making assumptions about the probabilistic state. In this sequential session a rewarding trace of a possible execution is computed which trades action costs, goal rewards, and determinacy. Taking the form of a classical plan, the trace specifies a sequence of actions that achieves the objectives following a deterministic approximation of the problem at hand. In our case this plan makes the assumptions that the room in the lower right of Fig. 4 is a kitchen and that the cornflakes can be found in that particular room. These assumptions are explicitly represented in the plan so that they can be partially revoked in case of a contingent planning session (cf. [2, in annex A.1]) of the switching planner. However, in the present case the assumptions made are indeed correct.

5. The sequential plan generated is now executed and takes Dora straight to the room that is (correctly) assumed to be a kitchen (place labelled

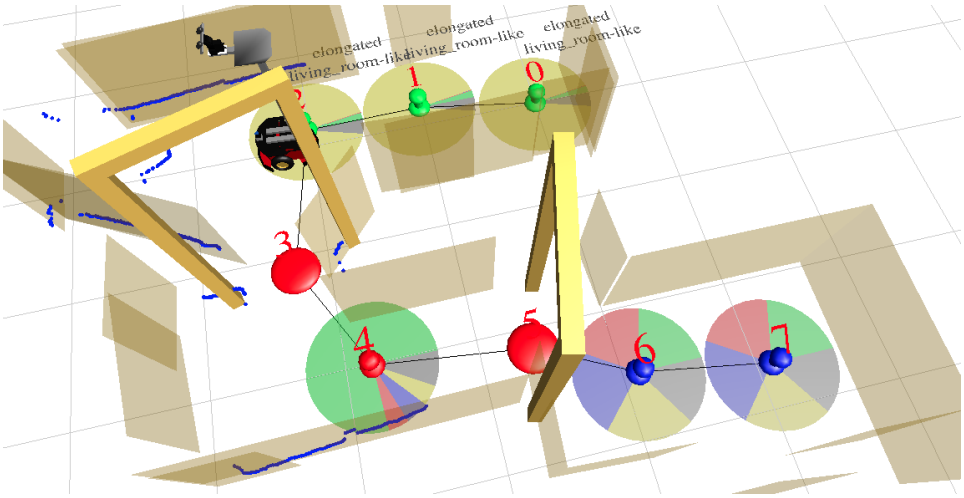



Figure 4: An exemplary representation in the moment the robot receives the command to find "cornflakes" in the real world home environment. The visualisation features numbered *places* and pie charts indicating probabilities of room categories (yellow=living room, red=kitchen, green=corridor, blue=office, grey=others). The labels attached to place nodes in the living room (upper right) indicate the most likely values in the distribution of classified *shape* (top) and *visual* (bottom) properties, respectively. Detected doors, used for room partitioning, are shown as door frames. The kitchen is at the lower right.

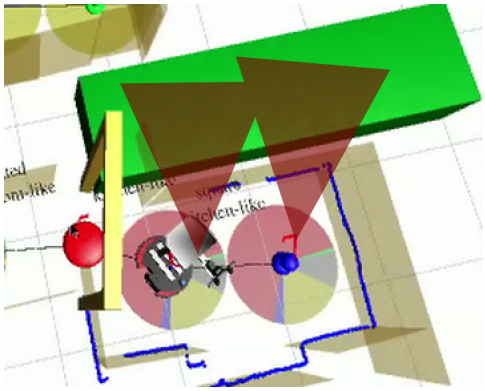


Figure 5: View cones generated trying to search for objects. This figure is just an illustration of two view cones, in reality there are more view cones generated the planner can then choose the best ones from.

6 in Fig. 4).

6. Once arrived at place 6 in our example Dora invokes the view point generation to determine promising views to find the object. In order to test the exploitation of on-ness relations in the system we pre-defined a table to be present in the kitchen. In Dora 2 this table surface is not recognised by the robot visually yet. The result of this processing step is a set of view cones oriented towards that table surface (illustrated in Fig. 5). Each view cones has attached the probability of finding the designated object in it, enabling the planner to choose among them intelligently.

7. Once the view cones have been generated, the switching planner enters a contingent session in order to do the observation planning. In this contingent session it reconsiders some of the assumptions made and also the observation models of its object detector, accounting for the uncertainty in perception and knowledge.

8. As part of the contingent session Dora in this exemplary course of action ran object detection in two of the view cones and found the object it searched for in the second.

9. In order to complete the task, which is not only to find the object but to also share the so far private belief about the object location with the human who was attributed the intention to know it, Dora heads back to the place she was at when she received the command and reports the position of the object again through the dialogue subsystem: "The cornflakes are in the kitchen at place 7."

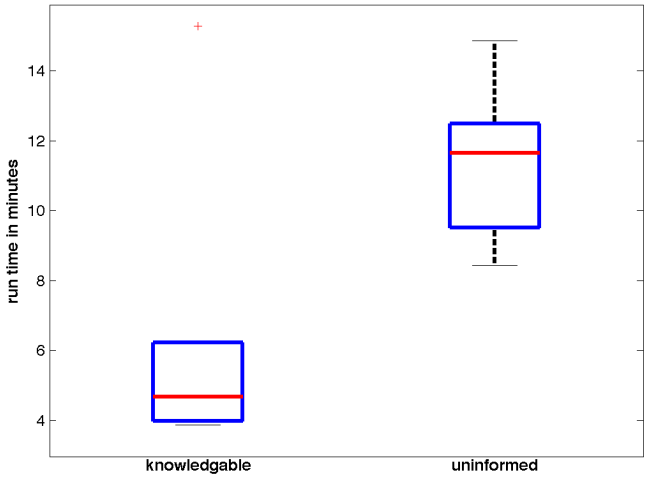
### 2.3.2 Comparison of full versus lesioned system

In order to test the effectiveness of (i) exploiting default probabilistic knowledge in a conceptual layer of our representation, (ii) the switching planner, and (iii) our implementation of the overall system, we ran two configurations ('full' and 'lesioned') of the system in the two environments a number of times and measured time to complete the task as well as success rates. Our evaluation compares the full system with a lesioned system in which the categorisation of visual appearance and shape properties has been disabled, emulating the limited reasoning capabilities available in our previous system Dora 1 [4, in annex A.2], where no such evidence was available.

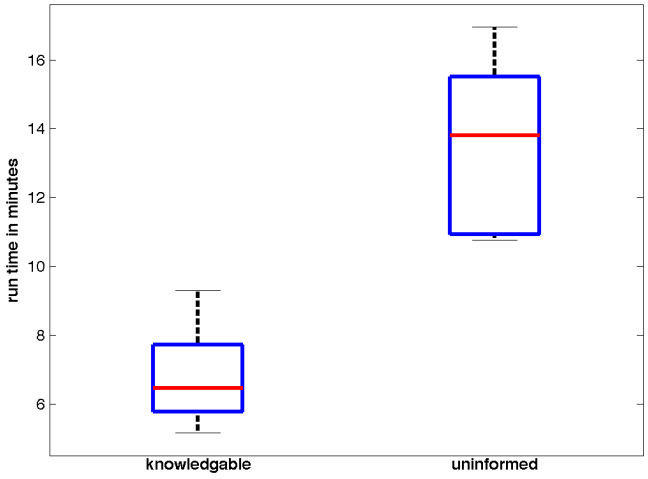
Fig. 6 shows the total runtime to complete the object search task in the lesioned (denoted as 'uninformed' in the figure) and the full system in both environments. What can clearly be seen from the figure is that the full system which can exploit the evidence about the categories of rooms achieves the task significantly faster (Mann-Whitney test $p < 0.01$ for both environments) on average. This is not a big surprise, as in these runs the system was facing what we call a *canonical* situation, in which the assumptions made by the planner in first place proved entirely correct during the execution of that plan. On the contrary, in the lesioned case the robot had less information and had to conduct a full exhaustive search. So it started its search in the living room because the object was as likely to be in this room than it was in any other.

However, an interesting case is represented by the outlier in Fig. 6(a). In this case the object to search was indeed in the kitchen, but the robot failed to detect it when it was executing its contingent plan due to recognition errors. As we added disconfirm and confirm actions to the planning domain, in this case the contingent session disconfirmed the existence of the object in this room and therefore handed control back to the sequential session of the planner, retracting the assumption that the cornflakes are in that room. As a consequence, the robot went back to the living room and performed another search in that room until it also (this time correctly) disconfirmed the existence of the object in it. At this point, there is still some probability mass left representing the likelihood of finding the object in the kitchen due to the fact that not all places have been searched during the first run. Also, the observation model of the object detector could account for the fact that we might just have missed the object in first place. In any case, the remaining probability mass causes the planner to make the assumption about finding the object in the room that actually is a kitchen again. This time Dora indeed finds the object. This case confirms the necessity and eligibility of a contingent planner in our system. It allowed the system to handle a perceptual error and still achieve the task.

We also tested the system in a non-canonical constellation. In this constellation the object was *not* located in the room that was a kitchen, but in




(a) BHAM.




(b) KTH.

Figure 6: Box and whisker diagrams of total runtime to achieve the given task in two environments comparing the 'knowledgable' or 'full' system to the 'uninformed' or 'lesioned' case.

the living room, hence, violating the default assumption about the location of objects. The behaviour observed was very much what we expected and quite similar to the outlier case discussed above: Dora started in the living room and went straight to the kitchen because the assumption was made to find the cornflakes in it. Again, the contingent planner drove Dora to search for the object until it was certain enough to disconfirm its existence in that room. Then, the next assumption made was that the object is in the living room, where in all three cases we tested, Dora eventually found the object and successfully accomplished its task.

## 2.4 Insights Gathered from Release 2

Integration for release 2 of Dora was identified with significant changes of the underlying representation to accommodate the uncertainty while still maintaining support for deterministic knowledge as well. This led to an incremental migration from one representation to another that is still reflected by the partially distributed state space mentioned earlier. Conceptually, this distribution is not problematic, but considering implementation it is still non-optimal and subject to further improvement.

CAST in general has proven to scale to more complex systems, with Dora 2 now being orchestrated of more than 90 components and a significantly increased coupling between these components. But managing this growing complexity also admittedly was a very demanding challenge. Distributed integration and flexible interaction pattern implemented through decoupled interaction on memory spaces, as put forward by CAST, impose issues such as synchronisation and implicit assumptions silently made by developers impeding the robustness of our systems. In order to conquer these challenges we already developed an analysis approach [14, in annex A.3] that helps us to gain more control and a better understanding of the system architectures.

Admittedly, the set of actions the planner can choose from is currently still quite limited and therefore it cannot yet play to its full strengths. In the current system, recognition of properties that are evident to room categories are acquired opportunistically, i.e. it is not planned for. For now, the only observation planning currently implemented in the system is on object detection.

Observation models for the object detector, occurrences of objects in rooms, and the occurrence of certain properties in dedicated types of rooms are all acquired offline from gathered data and assumed to be generic knowledge. However, a particular instance or situation the robot is in, e.g. a dedicated house, might not comply to the generic world assumption. In our current system, Dora is able to also achieve the task in such so-called non-canonical cases as has been proven when the object was located in the living room, but the robot does not yet *learn* from this experience for future

tasks.

Despite these limitations, this report demonstrates significant progress on a system level towards the CogX objectives of systems that self-understand and self-extend. The major contributions made enabled the system to cope with the uncertainty of knowledge and perception. The novel switching planner (WP 4) made it possible to handle hard decision-theoretic problems by partially revoking assumption made in a sequential session and consider alternatives. With the use of a chain graph model to reason in the conceptual layer of our representation (WP 3) we presented a coherent theory to combine instance and conceptual default knowledge, partially probabilistic and partially deterministic.

# 3 George

The main purpose of the George scenario is the development, demonstration and evaluation of our system for interactive learning in a dialogue with a tutor. The main goal this year was, therefore, to integrate the developed vision routines, learning and recognition competencies, dialogue capabilities, motivation and planning framework as well as different kinds of representations into an overall system.

## 3.1 Scenario

The robot operates in a table-top scenario, which involves a robot and a human tutor. The robot is asked to recognize and describe the objects in the scene (in terms of their properties like colour and shape). The scene contains a single object or several objects, with limited occlusion. The human positions new objects on the table and removes the objects from the table while being involved in a dialogue with the robot.

The robot has built-in abilities for communication with a human, visual processing, planning, etc. It also has built-in learning abilities, however it does not have any representations of objects or object properties given in advance. They are gradually built by observing objects in the scene and interacting with the human tutor in a mixed-initiative dialogue. Our objective is to demonstrate that the developed system can efficiently acquire conceptual models in an interactive learning process that is not overly taxing with respect to tutor supervision and is performed in an intuitive, user-friendly way.

We demonstrate these principles in the case of learning about object colours and basic shapes. We also (partially) integrated into the system the components for object recognition, affordance recognition, and high-level cross-modal learning. However, since they were not integrated into the main system, we did not evaluate them on the system level and we do not report this work in this deliverable.

## 3.2 Description of Release Yr 2

Our starting point was the George Year 1 system. The scenario setup and the main goals are similar to those we had in Year 1. However, we substantially improved the competencies of the Year 1 system and added new ones. Fig. 7 concisely depicts the main competencies of our system and the relationships between them, as well as the workpackages in which these competencies were developed. In short, by processing visual information (WP2 and WP5) and communicating with the human (WP6), the system forms beliefs about the world. They are exploited by the behaviour generation mechanism (WP1 and WP4) that selects the actions to be performed in order to extend the

system's knowledge about visual properties. In the following we first list the major improvements over the Y1 release and then focus on learning mechanisms that were in the focus of research in Year 2 in George scenario.

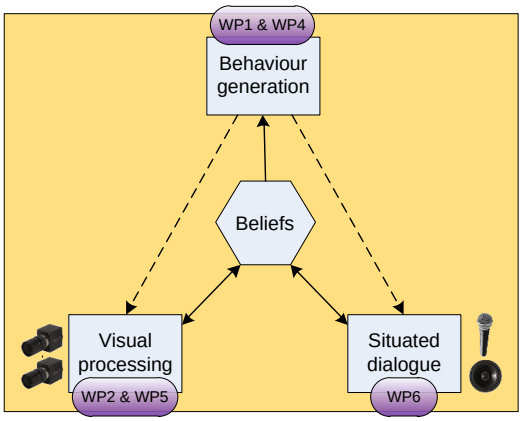



Figure 7: System competencies and relationships between them.

### 3.2.1 Major improvements

**Underlying vision.** When completing the integration of Year 1 system we found out that the underlying vision was not as robust as we wanted. Therefore we advanced the underlying computer vision routines to enable robust and persistent detection of objects in the scene (WP 2). We extended the detection of supporting planes to cover cases with multiple planes. To support increasingly dynamic scenes we added tracking mechanisms to maintain identity of proto-objects during short occlusions (caused e. g. by a manipulating hand) and during observer motion. We relaxed several assumptions and enable the learning framework to work in more realistic settings.

**Motivation and planning.** In the Year 1 system, the learning process was managed directly form Vision SA in a rather ad-hoc way. Now, we also integrated Planning SA in the system and the learning process is completely controlled by motivation (WP 1) and planning (WP 4) in a consistent way. Motivation mechanism monitors beliefs in the binder and generates and filters goals that are then used by the planner to determine which actions to execute in order to increase the knowledge about the objects. To support these processes we also integrated the new belief models and binding mechanism, as well as the new Dialogue SA (WP 6). We use very similar mechanisms as in Dora scenario, which will make a future merging of these two scenarios easier.

**Learning of object properties.** We improved the methods for learning object properties, which are now based on Discriminative Online Kernel Density Estimation (odKDE) [8] that we have developed (WP 5). The odKDE penalizes discrimination loss during compression of the generative models that it builds from the data stream, thus introducing a discriminative criterion function in the construction of generative models. As a consequence, the system builds the models of a lower complexity and achieves better recognition rate.

**Knowledge gap detection and learning mechanisms.** We implemented and integrated the knowledge gap detection methodology presented in WP 5, which detects the incompleteness in knowledge and provides the information needed by the planner to plan further actions. Using this methodology three major learning mechanisms have been implemented, which can be used in combination, i.e. in mixed-initiative learning. These learning mechanisms are presented in the next subsection.

### 3.2.2 Learning mechanisms

Fig. 8 depicts the schematic system architecture (here, the components are depicted as rounded boxes and exchanged data structures as rectangles, with arrows indicating a conceptual information flow). The system is composed of four subarchitectures, *Visual SA* implementing low-level vision routines and visual learning and recognition; *Dialogue SA*, implementing situated dialogue competence; *Binder SA*, storing the beliefs; and the *Planning SA*, implementing the goal generation and planning framework. The system is described a little bit more in detail in the attached Annex A.4 [18]; here we will only describe different learning mechanisms that drive the learning process in George and are also outlined in Fig. 8.

**Tutor-driven learning.** In the tutor-driven learning mechanism, the robot relies on the tutors initiative to provide information about the visible objects. The learning act occurs, when the visual subsystem detects an object and processes its visual features and the information provided by the tutor is successfully attributed to the same object. The motivation framework therefore monitors the Binder and when a new *attributed belief* is added, which carries information about an object, a goal is generated to update the representation with this information. The planner generates a trivial plan to updtate the visual representations; the execution subsystem delegates the visual learner in the visual subsystem to carry out this action. The tutor therefore drives the learning by providing the information to the tutor (e.g., T: "This object is red.").

**Situated tutor assisted learning.** In situated tutor-assisted learning the robot shows a greater degree of initiative. The robot can, depending on its current ability to recognise an object, ask a question about the object's properties. In this case, the motivation subsystem reacts to the *private belief*. The robot asks about the object property with the highest *information gain*, since it expects that the model of the corresponding object property will profit most if it gets the information it asks for. The information gain estimates how much the system would increase its knowledge, if it were to receive information from the tutor about the particular concept related to a particular object in the scene (e.g., the colour of the object). The planner generates a plan to ask questions about missing information. The execution subsystem generates a corresponding robot intention, which is further managed by the Dialogue SA, resulting in the synthesis of the corresponding generated utterance. Depending on the confidence in the recognition results the planner can select between polar questions (e.g., G: "Is the color of this object red?") and open questions when the recognition confidence is very low (e.g., G: "What is the color of this object?"). After the tutor provides the answer, the workflow is similar to the tutor-driven learning.

**Non-situated tutor assisted learning.** The robot's initiative goes even a step further in non-situated tutor-assisted learning. Here the robot also tries to influence the visual information it is getting by making requests to the tutor (e.g., G:"Could you show me something red?"). The *model status* (an epistemic structure similar to a belief) has a key role in deciding if and what kind of request to make. The model status contains key information about the visual models (knowledge) maintained by the visual learner. The most important information is again the *information gain*, which in this case estimates the reliability of a model in general, not relating this utility to a particular object in the scene. The goal generator that triggers this learning mechanism has the lowest priority and is usually triggered when no objects are present in the scene. Therefore, when the robot doesn't have anything else to do, it asks the tutor to show it an object with particular visual properties that would potentially increase the robot's models of these properties most.

## 3.3 Evaluation Approach and Framework

We evaluated the developed system in three ways: by running the system and interacting with it; by emulating the tutor and performing a systematic evaluation of the entire system; and by performing evaluation in a simulated environment. In the following we describe each of these approaches and briefly report the experimental results. The experiments are described more in detail in Annex A.4 [18].

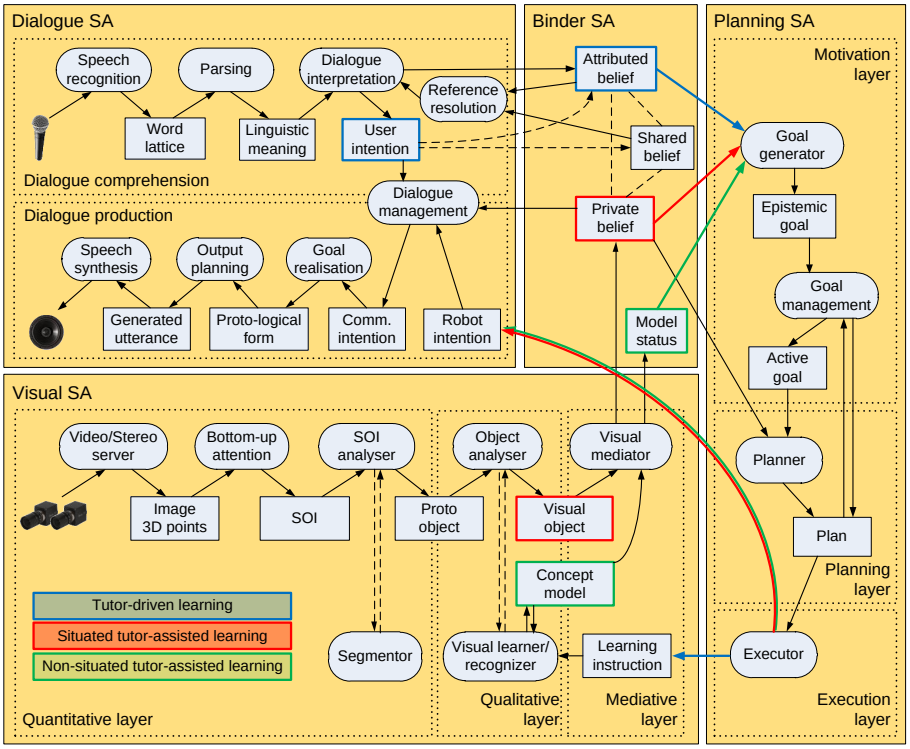



Figure 8: Schematic system architecture with indicated process flow for three learning mechanisms.

### 3.3.1 Interactive evaluation

The system has been designed to be used in interaction with a human tutor. We, therefore, engaged ourselves in dialogues with the robot and tested several aspects of the robot. By performing different actions we were trying to estimate where the limits of the system are. Such interactive work also helped us to update the requirements of the system and later to test and debug the developed system.

Fig.9 depicts fragments from one such run. The first fragment shows the dialogue at the beginning, when the system has not formed any model of any concept yet, so it cannot recognize the object properties. Then, in a mixed-initiative dialogue, the system first creates and then sequentially updates the models of the concepts that are being learned using the learning mechanisms presented above (tutor-driven learning, *TD*, situated tutor-assisted learning, *TA*, and non-situated tutor-assisted learning, *NS*). After a while George can successfully recognise the acquired concepts and provide reliable answers.

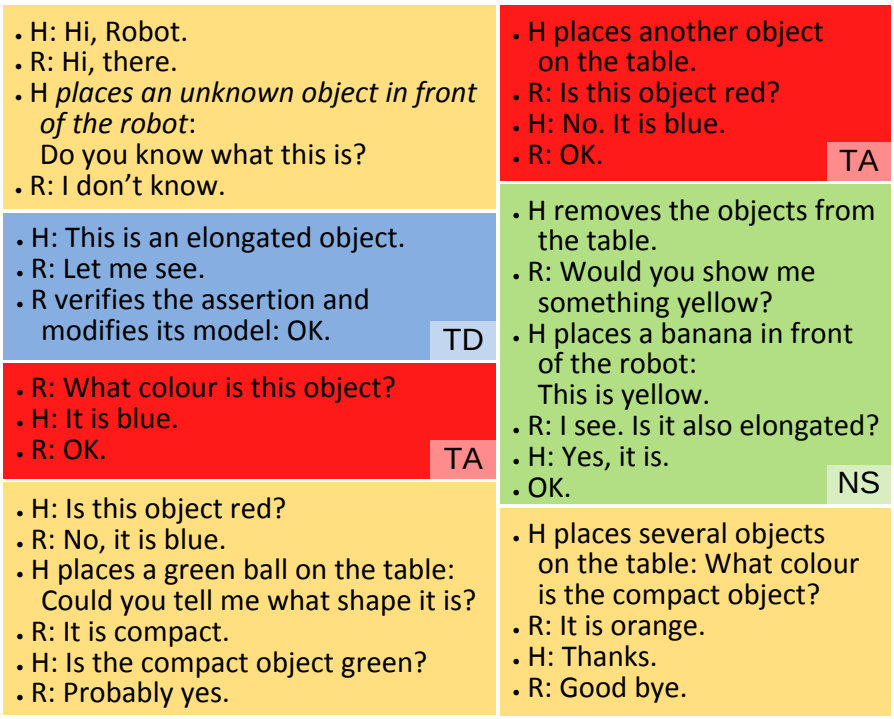



Figure 9: Fragments from a mixed-initiative dialogue.

### 3.3.2 System evaluation

The direct interactive work with the system is very time consuming and using this alone it is impractical to perform a thorough evaluation of the system. Learning is a long-term phenomenon; the system needs to observe a number of training samples to build a reliable and robust representation of the concept that is being modeled. And in order to obtain reliable results, the experiment has to be repeated several times. We therefore developed a component for automatic testing of the entire system.

We captured pairs of images and manually labelled them. Then, we replaced the image stream coming from the live cameras by reading these images from file. We also implemented a simple finite automata that emulated the tutor behaviour in the case of the tutor-driven learning; since the ground truth information about the visual properties of the objects was known, the emulated tutor could describe every image that was shown to the system. Apart from the camera input and speech recognition, the entire system worked in the same way as in the case of live operation, therefore we were able to evaluate the performance of the whole system.

The basic state machine is presented in Fig.10 (later it was extended, see Section 3.4). When the table is empty, the testing component loads the scene (i.e., loads the pair of images from the disc). Then it describes the object (by providing it the ground truth colour or shape label). When both visual attributes (colour and shape) are processed, the scene is unloaded and the entire cycle repeats.

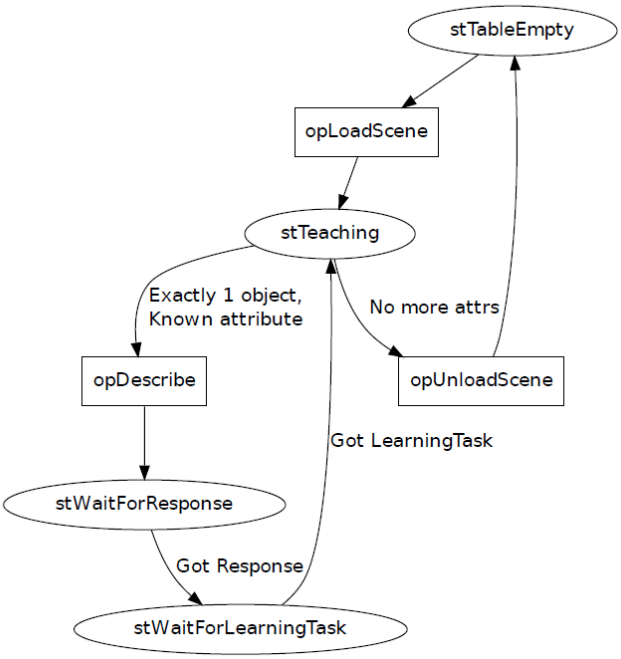



Figure 10: Basic state machine for the emulated tutor.

The extended version of this state machine was used for a systematic evaluation of the entire system. We collected a database of 1120 images of 129 objects. We used 500 pairs of images as training samples and the rest of them for testing the recognition performance. Eight colours and two basic shapes were being taught. After each update we evaluated the models by trying to recognize the colours of the objects in all test images. The model performance was evaluated in terms of recognition rate. We repeated the experiment three times by randomly splitting the set of images into training and test sets and averaged the results across all runs.

The experimental results are shown in Fig. 11. It shows the evolution of the learning performance over time. It is evident that the recognition rate improves with increasing numbers of observed images. The growth of the recognition rate is very rapid at the beginning when new models of newly introduced concepts are being added, and still remains positive even after

all models are formed due to refinement of the corresponding representations. These experimental results show that the entire system performs as expected; it is able to successfully detect the objects, understand the tutor describing these objects and build reliable models of visual properties.

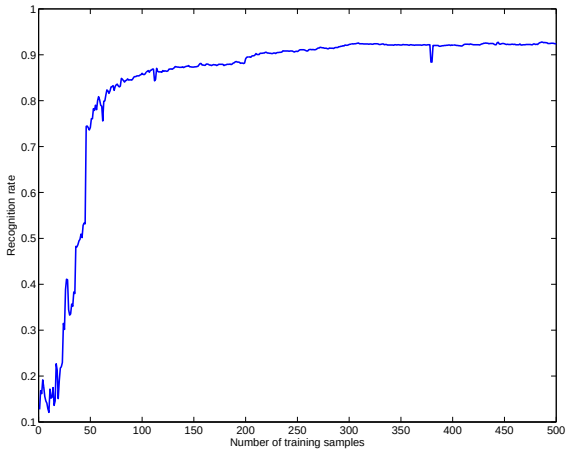



Figure 11: System evaluation - recognition rate.

### 3.3.3 Evaluation in simulated environment

To test the other learning mechanisms we would have to implement significantly more advanced tutor emulation (in fact, we would have to implement another Dialogue SA that would understand the robot's utterances), therefore we performed the evaluation of the proposed learning mechanisms in a simulated environment in Matlab.

We used the same set of 1120 pairs of images as in the previous experiment. We ran the visual subsystem of George, which was used for detecting and segmenting the objects. The extracted features were then used for evaluation of the learning mechanisms.

We tested the performance of all three learning mechanisms presented above ($TD$, $TA$, $NS$). In the tutor-driven case we also wanted to test the influence of order of training samples, so we evaluated two variants of the tutor-driven strategy; in the first case the training images were presented in a random order ($TD_{rnd}$), while in the second case the order was sequential ($TD_{seq}$).

We evaluated the performance of the learned models in terms of the recognition rate obtained on the training set. However, in such interactive learning settings, the success of recognition is not the only measure that matters. It is also very important how the learned models were obtained, i.e., how much effort the tutor had to invest in order to teach the robot. Measuring the tutoring cost in such a mixed-initiative learning framework is

quite a challenging problem; in this experiment we resorted to the following simple criterion: if the tutor had to provide the description of an object, it provided 3 bits of information (3 bits encode 8 classes of colours), while a polar answer was evaluated as a 1 bit cost. We therefore evaluated the different learning methods by comparing their recognition rate with respect to the cumulative tutoring costs. The evolution of the results over time is shown in Fig. 12.

The full description of the experiment is given in Annex A.4 [18]. Here we present just the most important results. The learning strategy $TA$ was the most successful in terms of reaching top performance with minimal information provided. The strategies $TD_{rnd}$, $TD_{seq}$ and $NS$ were equal in amount of information provided by the tutor, but there is a striking difference in the learning rate. We can see that the order in which the images were presented, played a very important role. When the images were presented in sequential order ($TDseq$), the learning progress was very slow, while learning with the random sequence ($TD_{rnd}$) lead to significantly better performance. The tutor, would therefore have to pay a lot of attention to which of the objects to present. The $NS$ approach achieved very similar results to the best $TD$ approach; in this case, however, the sequence of learning was dictated by the system, which would relieve the tutor. We can expect that by combining these learning strategies we could achieve even better results.

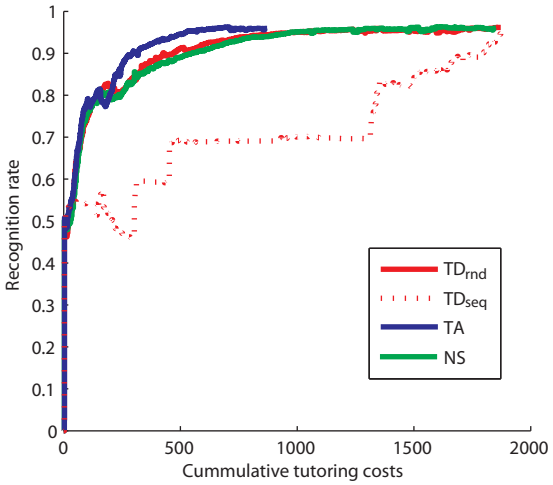



Figure 12: Evaluation of different learning strategies.

## 3.4 Insights Gathered from Release 2

George is a very complex system and interactive learning in dialogue with a tutor is a complex scenario, therefore there are many unexpected events

that can happen during operation. A human tutor can easily handle most of these exceptions, therefore in interactive work they do not tend to cause severe problems. The human tutor can, for instance, reposition the object if it was not detected, and repeat its description, or wait for the robot reply a little bit longer than usual. Since the system was designed to have a human in the loop, such problems are not critical.

When it comes to automatic evaluation of the system, these exceptions initially turned out to be a problem that prevented us to perform a thorough evaluation. The human tutor was in this case replaced with a rather simple finite automata, which cannot handle such exceptions. When we applied a simple state machine presented in Fig. 10, the system often stopped working as planned when an unexpected event happened. Therefore, we identified the problems and extended the state machine to detect and handle such events. The state machine of the enhanced emulated tutor that was used in the actual test is shown in Fig. 13. It handles the exceptions that the human would take into account in an interactive settings. This automata enabled us to perform thorough experiments as described in Section 3.3.2.

In our usual work with the robot we ran the full system for shorter periods of time, e.g., 15-20 minutes. To perform the evaluation of the learning mechanism we had to run the system for longer time. We successfully run the full system, observing the objects and learning their properties, for more than two hours. And since it is not the primary goal of the project to produce a system that would actively and robustly run for hours, we have not payed a special attention to the problems that arise in longer runs. Longer test runs therefore introduced new problems, mostly technical, but also some of conceptual nature. Until now we have not integrated into the system a variant of episodic memory. A cognitive architecture would from conceptual point of view need to support different levels and types of memory. When performing long test runs we also got a practical confirmation; the working memory entries that were not deleted, decreased the performance of the system and filled up the memory. If we want to run the system for long periods of time, we would have to implement a mechanism to abstract or/and move some of the content of the working memories to episodic-like-memory.

The current George system assumes a static robot platform and static cameras. This means that the active learning (as the central paradigm of the George scenario) is limited to the dialogue with the human tutor. The platform staticity also implies other limitations, like the single-view representation of the objects, while the robot has no notion of anything outside its static field of view (not visible is equivalent to not existing). In this sense the next logical step of the George scenario is the introduction of limited platform mobility, i.e. pan-tilt unit and limited movement around the table. This would extend the robot's awareness from the current field of view to all potentially perceivable environment (i.e. the whole table instead of just a part of it), introduce new possible actions for active learning (e.g.

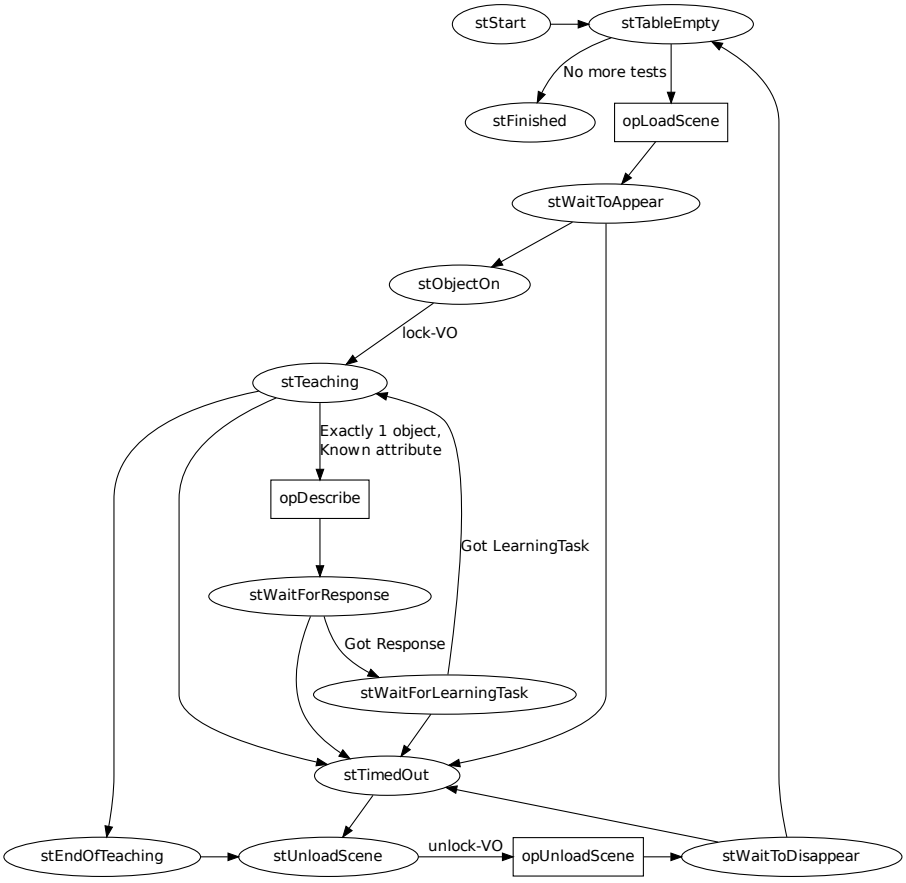



Figure 13: Complete state machine for the emulated tutor.

focus on object, look at it from another side, search the table for an object with particular properties, etc.) and extend the context in situated dialogue.

# 4 Dexter

Dexter implements scenarios that demonstrate the advances contributed by CogX partners in terms of manipulation under partial information.

Performing manipulation with partial information generally means that the effect on the agent's environment of even the most elementary actions is uncertain. Humans deal with this issue by performing manipulation within a tight sensorimotor loop. Within this loop, we seem to be able to predict the repercussions of manipulative actions onto sensory channels. This skill relies on a (learned) model of expected sensory feedback [6] which allows us to react to unexpected situations – e.g., slippage. The work lead at BHAM and KTH aims at endowing Dexter with such models (referred to as *forward* models below). Our aim is to provide our agent with means of learning what it feels like to manipulate an object, and what effect manipulative actions have on the kinematic behaviour of an object. During manipulation, sensory signals are compared to the signals predicted by the forward model. This comparison provides Dexter with means of detecting unexpected situations, possibly triggering corrections to the manipulation plan.

In this context, BHAM has developed a model that predicts the kinematic transformations that an object undergoes when pushed by a manipulator. For a given object configuration and a set of parameters defining a pushing action, the model provides the agent with the configuration in which the object will be after the push. In effect, this model works much like a kinematics simulator. However, contrary to classical simulators, the BHAM forward model is not parametrized by an expert. Instead, the agent learns model parameters from experience, by interacting with objects. This allows it to intimately capture the complex behaviours that result from the interactions of its manipulator and the world, and to capture object properties that would be difficult to measure otherwise (centre of mass, friction, etc.). Concurrently, KTH has developed models for predicting the stability of a grasp from the tactile feedback available to the robot before attempting to lift up and transport an object. By observing the tactile patterns that result from grasps applied onto different objects, our agent becomes increasingly efficient at predicting the tactile feedback that characterises robust grasps. Subsequently, when executing a grasp, afferent tactile signals are compared to those predicted by the learned model, yielding an estimate of the stability of the grasp.

To date, the behaviours made available by BHAM and KTH are implemented on two distinct platforms. Both are described below.

## 4.1 Object Behaviour under Pushing Actions (BHAM)

At BHAM we have been investigating the problem of learning to predict how objects behave under pushing manipulative actions. This enables us to

look at issues such as how the robot can learn about new objects, and how it can revise or add new models to its repertoire.

### 4.1.1 Scenario

Although pushing manipulation can be considered a simple experimental domain, yet in order to be able to plan and act in this domain, a robot must incorporate knowledge about various object properties which may or may not be observable. These properties can involve shape or frequently complex contact relations with other objects, but also mass, friction or elasticity.

We consider pushing manipulation as fundamental to grasping (in particular to in-hand manipulation) due to multiple (and variable) contacts between fingers and the manipulated object. This allows us to develop new prediction approaches which base on the work done so far in the pushing manipulation domain.

Most of the presented work has been published in two conference papers: learning predictive models in [7, in annex A.5] and visual tracking using predictive models in [13].

**What is new** As compared to the previous release, this year we made several advances as described in the paper that analyses the system. First we focussed on learning the predictors from real physical experiments. Second we introduced objects with more complex shapes and physical properties. Third a new experimental setup demonstrates an improved ability to generalise to novel shapes and novel pushing actions.

Importantly, we developed several new extensions to our prediction learning approach which greatly improve prediction performance and decrease the required minimum number of learning samples [7]. In particular, we introduced:

- A similarity transformation between local expert reference frames and the inertial frame.
- A new conditional kernel density estimator which uses quaternions and orientation statistics.

**Relation with other CogX systems** The prediction learning framework has been integrated with a visual tracking system [13] which has been also used in Dora scenario.

### 4.1.2 Evaluation approach

This deliverable is concerned with the analysis of the scenario systems. In the attached deliverable on Dexter we analysed the performance of our year 2 Dexter system in detail. We summarise this analysis here.

**Algorithms** We have investigated two groups of prediction algorithms, in which a robot learns to predict the object motions that will result from various applied pushes (for details see [7]).

1. In regression learning we used the LWPR regression learning technique [19] to learn the mapping between the current pose of an object, the planned push and the resultant pose of an abject.

2. A multiple experts approach also learns the mapping between current pose of an object, the planned push and the resultant pose of an abject. This mapping is approximated by a *global expert*. In addition, the approach takes into account information about local contacts between the robotic finger and the object (*local expert*) as well as contacts between the object and environment (*surface patch experts*).

   All experts are approximated by *conditional non-parametric density estimation* techniques similar to kernel density estimation.

The motion of the object as well as the motion of the object parts is represented by rigid body transformations where SO(3) rotation is parametrised by Euler angles. The most recent versions also use quaternions and von Mises-Fisher distribution to represent conditional distributions in our multiple expert-based approaches.

Furthermore, in multiple expert-based approaches we used a *similarity transformation* which effectively moves the global inertial frame to the instantaneous local frame of the object and the object parts (see [7]). Because the observed motion frequently has a simple structure as observed in the local frame, an appropriate choice of the local frame rotates the entire manifold so that it is aligned with one of the axes of the local coordinate frame. This allows us to approximate the full sample covariance by its diagonal elements only.

**Experiments** To test our prediction algorithms we performed several experiments, each consisting of multiple trials in which a robotic arm equipped with a finger performs a random pushing movement towards an object. A typical setup is shown in Figure 14. We used 5-axis robotic manipulator, and the motion of pushed objects is captured by a single camera and a visual tracking system [12].

We require the prediction algorithm to predict the entire object motion as a sequence of single-step predictions, before the physical push is initiated and without any correction from visual feedback during the push execution. We split experiments into three groups, where each group addresses a qualitatively different problem:

1. Learning to predict motion of real objects. We have shown that physics-based predictors (e.g. NVIDIA PhysX) are unable to ap-

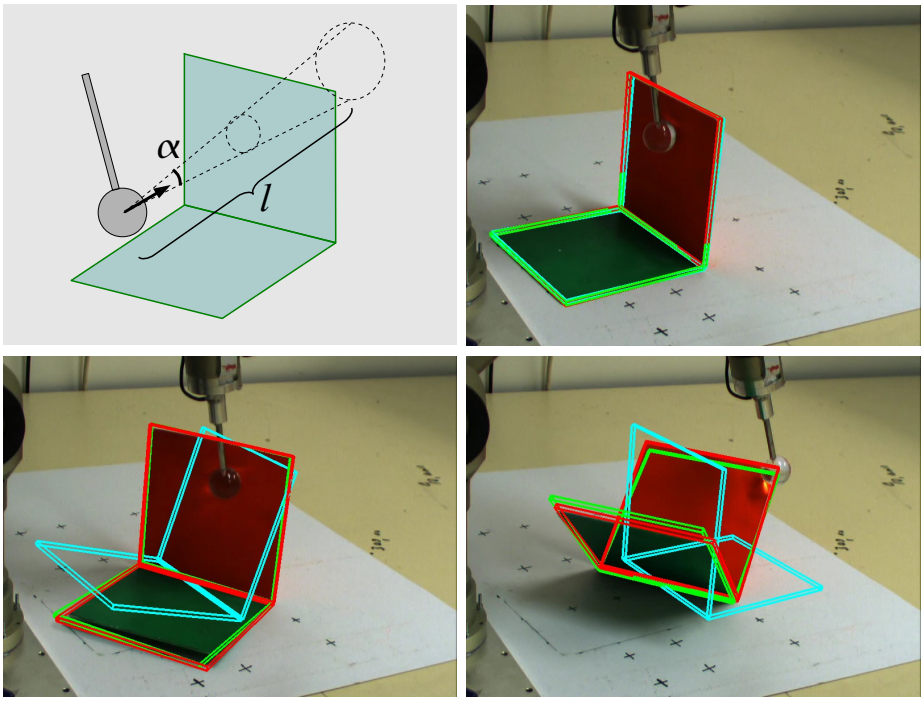



Figure 14: A 5-DOF robotic arm equipped with a finger performs a random straight-line pushing movement of a variable length $l$ within a cone with angle $\alpha$ towards an object (top left). Each single movement lasts 10 seconds, while data samples every 1/15th of a second. The object behaviour can be complex and varies depending on the finger trajectory and its pose relative to the object. In the image sequence shown above, the object begins to rotate anti-clockwise (top right - bottom left) before tilting (bottom right). The red wire-frame shows the output from the vision tracking system. The green wire-frame indicates the object pose predicted by the multiple-expert learning method, while the blue wire-frame is generated by the PhysX simulator.

proximate this motion as opposed to the introduced learning-based approaches.

2. Generalisation with respect to novel actions - learning to predict the object motion after applying novel pushes which has not been observed before. We have shown that the multiple expert approaches has clear advantage over the regression based ones. This is due to the fact that local experts and patch experts carry information which can be re-used in situations which are novel from a perspective of a gross motion of the object.

3. Generalisation with respect to novel shape of objects - learning to

predict the object motion which shape has not been observed before. Again, we have shown that the multiple expert approaches perform better than regression approaches. Local experts and patch experts prevent the multiple experts predictor from physically impossible motions which violate physics.

## 4.2 Tactile-based Grasp Stability Prediction (KTH)

The KTH Dexter platform consists in a Kuka arm and a Schunk hand equipped with tactile arrays. The KTH tactile-based stability model aims at predicting whether a grasp will be stable or unstable based on the stream of tactile data produced while the robot is closing its hand around an object, up until the hand is fully closed. The model is learned by executing sets of grasps on multiple objects and training stable vs. unstable classifiers on the collected data. By integrating tactile models with the pose tracker developed at TUW, our agent is also able to learn what objects should feel like *when grasped from a specific side*, which allows for further disambiguation of stable and unstable configurations. In a recent experiment, it was shown that the tactile-based stability model proved especially helpful when used in combination with a simulation-based grasp planner. Simulation-based grasp planners usually overlook many important object properties, such as friction or mass distribution. Moreover, plans can never be executed perfectly due to uncertainty in perceptual input (e.g., noise in a vision-based computation of an object's pose). The grasps suggested by simulation-based planners are thus often of uncertain practical usability. By combining a simulation-based planner with the tactile-based stability model, the agent is able to estimate, before lifting up an object, whether the object-gripper contacts that were actually achieved are likely to lead to a stable grasp. If the tactile information does not predict a stable grasp, the agent can adjust its grasping configuration or retract its manipulator and try a new grasping plan.

The work discussed above is well under way, and it will be included in the upcoming WP2 deliverables.

## References

[1] CogX Consortium. Deliverable DR.4.3: Planning for knowledge changes. Technical report, CogX, 2011. to be published in month 39.

[2] Marc Hanheide, Nick Hawes, Charles Gretton, Hendrik Zender, Andrzej Pronobis, Jeremy Wyatt, Moritz Göbelbecker, and Alper Aydemir. Exploiting Probabilistic Knowledge under Uncertain Sensing for Efficient Robot Behaviour. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 2011. accepted for publication.

[3] Marc Hanheide, Nick Hawes, Jeremy L Wyatt, Moritz Göbelbecker, Michael Brenner, Kristoffer Sjöö, Alper Aydemir, Patric Jensfelt, Hendrik Zender, and Geert-Jan M Kruijff. A Framework for Goal Generation and Management. In *Proceedings of the AAAI Workshop on Goal-Directed Autonomy*, 2010.

[4] Nick Hawes, Marc Hanheide, Jack Hargreaves, Ben Page, Hendrik Zender, and Patric Jensfelt. Home Alone : Autonomous Extension and Correction of Spatial Representations. In *Proc. Int. Conf. on Robotics and Automation*, 2011.

[5] Nick Hawes and Jeremy L Wyatt. Engineering intelligent information-processing systems with CAST. *Adv. Eng. Inform.*, 24(1):27–39, 2010.

[6] R.S. Johansson. Sensory input and control of grip. In *Novartis Foundation Symposium*, pages 45–58, 1998.

[7] Marek Kopicki, Sebastian Zurek, Rustam Stolkin, Thomas Mörwald, and Jeremy Wyatt. Learning to predict how rigid objects behave under simple manipulation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA11)*, May 2011.

[8] M. Kristan and A. Leonardis. Online discriminative kernel density estimation. In *International Conference on Pattern Recognition*, pages 581–584, Istanbul, Turkey, 23-26 August 2010.

[9] S. L. Lauritzen and T. S. Richardson. Chain graph models and their causal interpretations. *J. Roy. Statistical Society, Series B*, 64(3):321–348, 2002.

[10] J. M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *J. Mach. Learn. Res.*, 11:2169–2173, August 2010.

[11] T. Mörwald, J. Prankl, A. Richtsfeld, M. Zillich, and M. Vincze. BLORT – The Blocks World Robotic Vision Toolbox. In *Proc. ICRA*

*Workshop Best Practice in 3D Perception and Modeling for Mobile Manipulation*, 2010.

[12] T. Mörwald, M. Zillich, and M. Vincze. Edge tracking of textured objects with a recursive particle filter. In *Proceedings of the Graphicon 2009*, Moscow, Russia, 2009.

[13] Thomas Mörwald, Marek Kopicki, Rustam Stolkin, Jeremy Wyatt, Sebastian Zurek, Michael Zillich, and Markus Vincze. Predicting the unobservable, visual 3d tracking with a probabilistic motion model. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA11)*, May 2011.

[14] Jens Otto, Sebastian Wrede, Marc Hanheide, and Nick Hawes. A Unifying Model-Centric Analysis Approach for Robotic Systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011. submitted.

[15] A. Pronobis, O. M. Mozos, B. Caputo, and P. Jensfelt. Multi-modal semantic place classification. *Int. J. Robot. Res.*, 29(2-3):298–320, February 2010.

[16] K. Sjöö, A. Aydemir, D. Schlyter, and P. Jensfelt. Topological spatial relations for active visual search. Technical Report TRITA-CSC-CV 2010:2 CVAP317, Centre for Autonomous Systems, KTH, Stockholm, July 2010.

[17] Kristoffer Sjöö, Alper Aydemir, T. Mörwald, K. Zhou, and Patric Jensfelt. Mechanical support as a spatial abstraction for mobile robots. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2010.

[18] Danijel Skočaj, Matej Kristan, Alen Vrečko, Marko Mahnič, Miroslav Janíček, Geert-Jan M. Kruijff, Marc Hanheide, Nick Hawes, Thomas Keller, Michael Zillich, and Kai Zhou. A system for interactive learning in dialogue with a tutor. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011. submitted.

[19] S. Vijayakumar, A. D'souza, and S. Schaal. Incremental online learning in high dimensions. *Neural Computation*, 17(12):2602–2634, 2005.

# A Annexes

## A.1 Exploiting Probabilistic Knowledge under Uncertain Sensing for Efficient Robot Behaviour

**Bibliography** Marc Hanheide, Nick Hawes, Charles Gretton, Hendrik Zender, Andrzej Pronobis, Jeremy Wyatt, Moritz Göbelbecker, and Alper Aydemir. Exploiting Probabilistic Knowledge under Uncertain Sensing for Efficient Robot Behaviour. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 2011. accepted for publication

**Abstract** We present a mobile robot system that showcases two original contributions addressing uncertainty. Our first contribution is a probabilistic model for representing sensed and common-sense properties of space. Our second contribution is a continual planning system which is able to plan in large problems posed according to that model, by automatically switching between using decision-theoretic and classical procedures. We evaluate our system by having it perform object search tasks in two different real-world indoor environments. By reasoning about the trade-offs between possible courses of action with different informational effects, and exploiting the cues and general structures of those environments, our robot is able to consistently demonstrate efficient and robust goal-directed behaviours.

**Relation to WP** This paper is the major conference paper reporting the system evaluation carried out in the Dora scenario. It presents the *switching planner*, being the major contribution in WP4, the new conceptual layer of the spatial representation as a contribution from WP1 and WP3, and the overall integration and evaluation efforts being part of WP7. Integrated works from WP2 and WP6 are briefly mentioned as well. The evaluation reported was carried out in a real world environment of an inhabited house.

# Exploiting Probabilistic Knowledge under Uncertain Sensing for Efficient Robot Behaviour

**omitted for review**

## Abstract

We present a mobile robot system that showcases two original contributions addressing uncertainty. Our first contribution is a probabilistic model for representing sensed and common-sense properties of space. Our second contribution is a continual planning system which is able to plan in large problems posed according to that model, by automatically switching between using decision-theoretic and classical procedures. We evaluate our system by having it perform object search tasks in two different real-world indoor environments. By reasoning about the trade-offs between possible courses of action with different informational effects, and exploiting the cues and general structures of those environments, our robot is able to consistently demonstrate efficient and robust goal-directed behaviours.

## 1 Introduction

One dream of the AI community is to build a robot capable of performing tasks on demand in dynamic real-world environments like homes and offices. Such a robot must perform task and observation planning under uncertainty in pursuit of its current goals. It must do this whilst exploiting knowledge about the nature of the environments in which it is expected to operate. Towards realising the stated dream, this paper presents a robot system that uses a new planning approach to reason with new representations of space. Our approach integrates probabilistic models of background conceptual knowledge, and the visual appearance of objects and of room categories, to solve an object search task. These are used to create and maintain a probability distribution over possible maps that respect explicit representations of qualitative spatial structure. In order to allow the robot to effectively exploit this knowledge, we have developed a novel system for continual planning that automatically switches between using decision-theoretic and classical procedures to synthesise efficient action strategies.

We have implemented our approach on the mobile robot depicted in Fig. 1, and evaluated that system by having it perform object search tasks in real-world home and office environments. The objects it is able to search for are all *instances* of *categories*, e.g. a specific box of cornflakes in the kitchen is an instance of the category of cornflakes boxes, which is itself a sub-category of cereal boxes. The robot uses highly structured representations of knowledge at this *conceptual* level – e.g. cereal boxes are often located in kitchens or dining rooms, and sofas are often located in sitting rooms. Such *relational structure* expresses generalisations across multiple environments, and can be naturally represented probabilistically in order to support intelligent decision making across multiple environments. For our work, in an offline step we have compiled a knowledge base that captures that common-sense knowledge. Our two key novel contributions are:

Figure 1: Two exemplars of our robot system in the home and office environments they were tested in. The robots are equipped with a laser scanner and a stereo camera rig.

**1. A chain graph representation** that combines general purpose and contingent spatial knowledge in a single structure, together with processes for creating and maintaining that graph. This structure models the (uncertain) contingent knowledge the robot has about instances (e.g. what category of room it thinks room 1 is) in conjunction with its (also uncertain) long term conceptual knowledge (e.g. which types of objects are located in a particular category of room).

**2. A switching continual planner** that synthesises action strategies for very large partially observable decision processes posed by the tasks we consider. Our approach is to switch between decision-theoretic and classical modes of planning at different levels of abstraction. The classical system quickly solves a determinisation of the problem at hand, interpreting probabilistic information in terms of a cost model. The decision-theoretic system quickly solves abstract decision problems posed according to the current serial plan, and underlying belief-state. Overall, this approach allows the system to exploit our rich representation of spacial knowledge, and generate intelligent behaviour under uncertainty in a timely manner.

## 2 Related work

Probabilistic representations are employed for many specific, localised functions in robots operating in the real-world. For example, recent systems by [Gross *et al.*, 2009] and [Maier and Steinbach, 2010] employ probabilistic frameworks to model human interaction partners or visual appearance respectively. Other systems employ probabilistic representations across more subsystems but in limited ways. For example, [Kraft *et al.*, 2008] treat sensing deterministically and beliefs qualitatively during planning. We are not aware of any related system that features both a unifying probabilistic representation, and a domain-independent planner which is able to reason quickly over that unified decision-theoretic model to generate robot behaviour.

Object search with mobile robots has been studied for almost 20 years [Shubina and Tsotsos, 2010], yet no previous system reasons with uncertain conceptual knowledge about room and object categories. Instead, most dedicated systems treat the problem as a geometric one. Closest to our approach is the work by [Sjöö *et al.*, 2010b] who used common-sense knowledge encoded into a rule-based ontology to inform a deterministic planner which previously categorised room to search for a particular object. [Galindo *et al.*, 2005] make this relationship bi-directional: objects define room categories, while room categories provide information on where objects may be found. In [Bouguerra *et al.*, 2007] this approach was extended to treat some of the conceptual knowledge as uncertain, although restricted to the number of occurrences of object types in rooms. [Vasudevan and Siegwart, 2008] went beyond this to perform room categorisation through Bayesian reasoning about the presence of objects, but did not (as none of these did) include observation models in their reasoning (thus perception was still considered to be deterministic).

## 3 System Overview

Driven by the need to exploit conceptual knowledge for more efficient object search, our robot specifically employs processes to perceive instance information which it can relate to known concepts. The creation of a spatial model (containing rooms, doors, corridors etc.) is supported by metric and topological mapping processes. Room categorisation is supported by object detection (from pre-trained models) plus visual and geometric feature extraction. These processes are shown at the bottom of Fig. 2. They feed into the *conceptual layer* of a spatial representation designed following the principles proposed by [Pronobis *et al.*, 2010]. This layer maintains *instances* of predefined *concepts* and the *relations* between them. This conceptual layer not only contains the instance knowledge acquired through sensing, but also the *conceptual knowledge* we aim to exploit to generate efficient behaviour.

The system as described so far is similar to its predecessor [X, 20xx]. This previous robot was able to exploit *deterministic* conceptual and instance knowledge to achieve goal-directed behaviour and also perform object search. However, that system was unable to systematically cope with the uncertainty of sensing and the probabilistic nature of conceptual knowledge. For example, the deterministic knowledge that cornflakes are found in kitchens would cause that robot

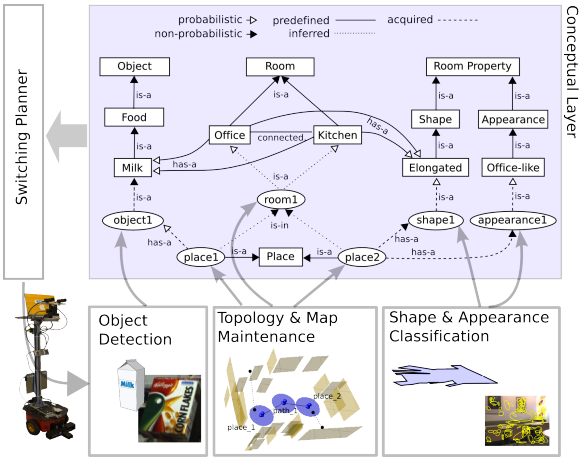



Figure 2: An abstract view of the processes and representations of the system. Sensing processes (at the bottom) discretise and categorise sensor input into instances and acquired relations in conceptual layer. This layer also comprises conceptual and ontological knowledge represented as a probabilistic graphical model of which only an excerpt in shown. The switching planner reasons upon the state distribution defined by the conceptual layer.

to search only in rooms that have been determined to be kitchens. It did not allow the robot to account for the likelihood that the cornflakes might also be in the living room, or even that it was simply unable to infer the category of a room correctly. We have overcome this limitation by employing a Bayesian approach within the conceptual layer in order to combine representations of certain knowledge (e.g. that cornflakes boxes are a type of cereal box) and also uncertain knowledge (e.g. that cereal boxes are located in kitchens $34\%$ of the time) in a unified framework. The style of arrows in the conceptual layer in Fig. 2 indicate the different types of relations. The conceptual layer is described in more detail in Sec. 4. It provides the state distribution on which planning is based. The planner we developed to meet the constraints imposed by the probabilistic representation is described in Sec. 5.

### 3.1 Sensing & Acting

In our system sensing is managed by a collection of processes which abstract from odometry data, laser scans and video sequences to maintain *instances* and the *probabilistic relations* which link these instances to concepts and other *instances*. We distinguish *continuous* and *active* sensing. The former is passive, continuously managing certain beliefs about the world. It is lightweight, and does not require a planner that might schedule information gathering actions. In contrast, active sensing is deliberately planned for.

**Mapping and topology maintenance** is a continuous sensing process that uses a SLAM algorithm [Folkesson *et al.*, 2007] to maintain metric and topological maps of the environment. It continuously localises the robot in those map. It discretises space into metrically localised *places* approximately $1m$ apart (represented by discs in Fig. 3). It also maintains a navigation graph that supports movement from one place to another. Place existence and connectivity is treated deterministically in our current system. In order that topological

places be interpreted with respect to higher level spacial concepts, mapping also features door frame detection from laser data. Doors are used to group places into instances of rooms following our non-monotonic reasoning approach [X, 20xx]. The results of this continuously running process are instances of places and rooms with acquired connectivity relations.

**Shape and appearance classification** is a continuous sensing process which generates properties characterising the shape and visual appearance of rooms. It extracts simple geometrical features from laser scans and uses Composed Receptive Field Histograms from acquired camera images as visual features [Pronobis *et al.*, 2010]. These features are evaluated on the basis of Support Vector Machine (SVM) models representing specialised concepts of "Room Property" (e.g. elongated, office-like etc.). Accumulated confidence gained from the SVM models for all views obtained at a topological *place* are normalised to gain probabilities. These are represented in the probabilistic "is-a" relation that ties property instances to the specific concepts (cf. Fig. 2).

**Object detection** is the only active sensing process in our system. It is triggered as part of the active search strategy described in the next paragraph. Objects are detected using the BLORT 3D object recogniser [Mörwald *et al.*, 2010] applied to images from one of the robot's cameras. This detector is characterised by false positive and false negative detection rates that characterise the observation model of this sensing action. Observation models allow the robot to reason that it might not have sensed an object despite it being perceivable, and vice-versa. This allows us to quantify the effects of active sensing processes on our conceptual knowledge. A detected object leads to the creation of a "has-object" relation for the specific instance the robot was looking for (cf. Fig. 2).

**Actions** in our system are all triggered by the planner. The planner typically solves two sub-problems: *navigation* and *local active visual search*. Navigation in the world is planned using the navigation graph defined by the connectivity relations. Movement between places is executed by the navigation component and includes local object avoidance. Local active visual search first requires an action to trigger the generation of discrete *viewpoints*. Following [Sjöö *et al.*, 2010a], the generation action is executed as a Monte-Carlo sampling of local metric maps yielding information about the probability of object presence. Viewpoints are assigned an observation probability for a set of objects. The planner then reasons using actions to move to a viewpoint and trigger object detection for appropriate objects.

## 4 Probabilistic Conceptual Representation

The conceptual layer is the main novel spatial development in this work. Implemented above the qualitative spatial framework in [X, 20xx], this layer facilitates inference that accounts for general high-level taxonomic conceptual knowledge, and the outputs of sensing. The former corresponds to a rich probabilistic ontology that specifies categories of room (kitchen, office, etc), how rooms are composed of topological places, and the likelihood that certain types of rooms will contain particular objects. The outputs of sensing include the results of shape and appearance classification, and also door and object detections.

**Chain graph model** All higher level inference performed by the robot is done according to a unified model of the conceptual knowledge and of the robot's sensing capabilities. That unified model is expressed using a *chain graph* [Lauritzen and Richardson, 2002], whose structure is adapted online according to the state of underlying topological map. Chain graphs provide a natural generalisation of directed (Bayesian Networks) and undirected (Markov Random Fields) graphical models, allowing us to model both "directed" causal (such as "is-a" relations) as well as "undirected" symmetric or associative relations (such as connectivity). In particular, our graph allows the robot to perform inference related to the ontology presented in Fig. 2, in addition to belief-state estimation from sensing. For planning, the chain graph is the sole source of belief-state information. In the chain graph, belief updates are event-driven. For example, if an appearance property, or object detection, alters the probability of a relation, inference proceeds to propagate the consequences throughout the graph. In our work, the underlying inference is approximate, and uses the fast Loopy Belief Propagation [Mooij, 2010] procedure.

Structurally, the graph encapsulates a taxonomy of concepts through hyponym relationships (is-a) as well as relations between rooms and objects (has-a relationships). It distinguishes three primary sources of knowledge: (1) predefined conceptual knowledge, e.g. the likelihood that cornflakes occur in kitchens; (2) knowledge acquired directly according to sensing, e.g. the likelihood of being located adjacent cornflakes given the result of an object detection; and (3) inferred knowledge, e.g. that the room is likely to be a kitchen, because you are likely to have observed cornflakes in it.

**Quantifying relations** It is at the core of our approach that some classes of representations are uncertain, while others are deterministic. We quantify the uncertainty in a Bayesian framework. Probabilities for acquired relations (cf. Fig. 2) are assigned by sensing processes. For example, when an object is sensed it is tied to a distribution of places where it might be located via a "has-object" relation.

It is the common-sense conceptual knowledge that allows the robot to be efficient in different environments. We want to endow the robot with knowledge about how likely it is that different objects will be found in different kinds of indoor locations, how shape and appearance properties relate to those locations, and how rooms of different types are typically connected. We do this by quantifying probabilistic relations as defined in the conceptual layer (cf. Fig. 2). Ideally these relations should be quantified by appropriate AI techniques that estimate the distribution as a generic model (although they could be prescribed in an ad-hoc fashion if the probability table is small and obvious). In our system, the "has-object" relation for rooms and objects demanded a principled approach to quantification due the combinatorial complexity of object-location cooccurrence applied to real world ontologies. To achieve this we applied an approach that exploited common-sense knowledge available through the world wide

web. First, the "has-a" relation linking rooms and objects was bootstrapped using a part of the *Open Mind Indoor Common Sense* database[1]. This contains a list of user-given associations between common everyday objects and their typical locations, providing us with a set of 5,800 deterministic relations between locations and objects. In order to quantify these relations we then used the object-location pairs to generate '*obj* in the *loc*' queries to an online image search engine. The number of hits returned was then correlated with the total number of hits for querying '*loc*' images, yielding a co-occurrence frequency distribution of approximately 1.5 million object-location pairs.

## 5 Switching Continual Planner

To generate flexible goal-oriented behaviour our system employs a domain-independent planner. From a planning perspective, mobile robots pose important and contrary challenges. On the one hand, planning and execution monitoring must be lightweight, robust, timely, and should span the lifetime of the robot. Those processes must seamlessly accommodate exogenous events, changing objectives, and the underlying *unpredictability* of the environment. On the other hand, in order to act intelligently the agent must perform computationally expensive reasoning about *contingencies*, and possible revisions of subjective belief according to quantitatively modelled uncertainty in acting and sensing. Addressing specifically this second challenge, [Talamadupula *et al.*, 2010] identify *continual planning* in the presence of detailed probabilistic models as an important direction for future research.

There has been much recent work on scaling POMDP solution procedures to medium-sized instances. In the case of general domain-independent factored systems, the state-of-the-art scales to relatively small problems with $2^{22}$ states [Shani *et al.*, 2008].[2] At their limit, these procedures take over an hour to converge. For classes of POMDP that feature exploitable structures (e.g., no actions with negative effects), problems with as many as $10^{30}$ states can be targeted by offline procedures [Brunskill and Russell, 2010]. Moving someway towards addressing all the challenges we have outlined, recent online POMDP solution procedures have been developed which can exploit highly approximate value functions – typically computed using a point-based procedure – and heuristic in forward search [Ross *et al.*, 2008]. Their applicability in our setting is limited due to the large amount of *problem-specific* offline processing required to get useful search guidance. A *very* recent and promising online approach for large POMDPs employs Monte-Carlo sampling to break the curse of dimensionality in situations where goal reachability is easy [Silver and Veness, 2010]. Although we suppose it an interesting item for future work to pursue that direction, it should be noted that ease of goal reachability is not guaranteed in the problems we face, and is certainly not a property to be assumed in domain independent planning.

In our work we take a concrete step towards addressing all the challenges we outlined. We have developed a *switching* domain-independent planning system that operates according to the continual planning paradigm. It uses first-order declarative problem and domain representations, expressed in a novel extension of PPDDL [Younes *et al.*, 2005] called *Decision-Theoretic (DT)PDDL*, for modelling stochastic decision problems that feature partial observability. The system *switches*, in the sense that the underlying planning procedure changes depending on our robot's subjective degrees of belief, and progress in plan execution. When the underlying planner is a deterministic sequential planner, i.e., a *classical* planner, we say planning is in a *sequential* session, and otherwise it is in a *contingent* session. Finally, planning is continual in the usual sense that, whatever the session, plans are adapted and rebuilt online in reaction to changes to the planning model (e.g. when objectives are modified, or when our robot's path is obstructed by a door being closed). By autonomously mixing these two types of sessions our robot is able to be robust and responsive to changes in its environment *and* make appropriate decisions in the face of uncertainty.

During a sequential session, a rewarding *trace* of a possible execution is computed using a modified version of the cost-optimising satisficing planner *Fast Downward* [Helmert, 2006] which trades action costs, goal rewards, and determinacy. Taking the form of a classical plan, the trace specifies a sequence of actions that achieves the objectives following a deterministic approximation of the problem at hand, i.e., a *determinisation* [Yoon *et al.*, 2007]. Structurally, it is a sequence of elements that are either: (i) actions from the DTPDDL description of the world, or (ii) atomic *assumptions*, modelled as deterministic actions, made about the truth value of facts that can only be determined at runtime (e.g., that a box of cornflakes is located on the corner bench in the kitchen). The system always begins with a sequential session, and once *Fast Downward* produces a trace, plan execution proceeds by applying actions from that trace in sequence until the applicability of the next scheduled action is too uncertain according to a threshold parameter (here, set at 95%). A contingent session then begins which tailors sensory processing to determine whether the assumptions made in the trace hold, or which otherwise acts to achieve the overall objectives.

Because decision-theoretic planning in large problems is too slow for our purpose (we seek response times in seconds), contingent sessions plan in an abstract decision process determined by the current trace and underlying belief-state. This abstraction is constructed by first excluding all propositions that are not true of any state in the trace, then adding them back, using as a heuristic the entropy of the trace assumptions conditional on a candidate proposition. Propositions are added, one at a time, until the number of states in the initial belief-state reaches a given threshold (here, 150 states). To the resulting abstract model we also add *disconfirm* and *confirm* actions that the contingent session can schedule in order to judge an atomic assumption in the trace. In the abstract model these actions yield a small reward if the corresponding

[1] `http://openmind.hri-us.com/`

[2] Considering only room categories and distribution of objects, problems we consider in this paper have $\sim 10^{27}$ states. The details of view points, from local active visual search, and those of robot location, further increase that figure. Therefore, not only because they are offline, but also because they have limited scalability, these approaches are infeasible in our setting.

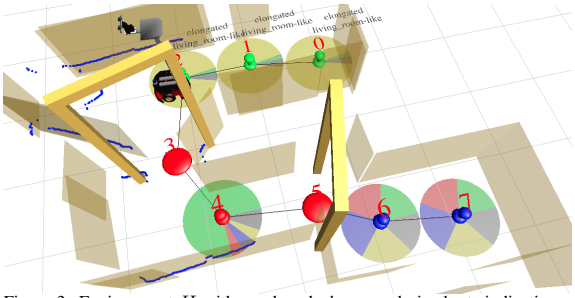



Figure 3: Environment $H$ with numbered *places*, and pie charts indicating probabilities of room categories (yellow=living room, red=kitchen, green=corridor, blue=office, grey=others). The labels attached to place node in the living room (upper right) indicate the most likely values in the distribution of classified *shape* (top) and *visual* (bottom) properties, respectively. Detected doors, used for room partitioning, are shown as door frames. The kitchen is at the lower right.

judgement is true (or small penalty otherwise). Once a judgement action is scheduled for execution the contingent session is terminated, and control is returned to a sequential session.

## 6 Real World Experiments

To evaluate the implemented representational and planning techniques, we analysed our robot system performing an object search task in two different environments: an larger office ($O$, 13 places in 3 rooms) and a smaller home ($H$, 7 places in 3 rooms). The acquired map for $H$ is shown in Fig. 3. Our evaluation compares the full system described in this paper, exploiting probabilistic conceptual knowledge and linked to sensing, to a baseline system that cannot make use of the conceptual knowledge. We refer to these as the "full" and "lesioned" systems respectively. In the lesioned system continuous sensing of shape and appearance properties is disabled, emulating the limited reasoning available in our previous system [X, 20xx]. Therefore it cannot use these properties to access the conceptual knowledge about categories of rooms encoded in the chain graph model, or the conceptual knowledge about object-location co-occurrence.

In all runs, a box of cornflakes (the object to search for) was placed in the environment among many other objects belonging to the nine categories the robot has been trained to detect. In the experiments, only a subset of all object-location co-occurrence frequencies consisting of 152 relations between the 19 selected object concepts (cornflakes among them) and seven given room concepts (among them kitchen, living room, corridor, and office) was employed. In a first set of runs directly comparing the full system ($FC$) to the lesioned case ($LC$), the box of cornflakes was in the kitchen, which is the canonical location for this type of objects according to the common-sense conceptual knowledge (with a probability of $P(cornflakes|kitchen) = 0.336$). In a second set of runs, the object was at a non-canonical location ($P(cornflakes|living\_room) = 0.035$) to test the full configuration (results denoted as $FNC$).

**Hypotheses** The hypotheses leading to this study design are that (i) the exploitation of the conceptual knowledge in the full system will enable the robot to achieve the task quicker in canonical cases when compared to the lesioned system in the same experimental setup, (ii) although more efficient in the average case, the system will be robust in the presence of sensing errors, and (iii) that the system will still be able to achieve its goal, even relatively efficiently, in non-canonical setups. In all runs, before the robot was given the goal to find the object, it performed a short exploration of adjacent places to sense room properties in order to infer the category of the room (if this evidence was not lesioned).

| conf. | obj. loc. | lesion | ♯succ./♯tot. $H$ | ♯succ./♯tot. $O$ | avg. time $H$ | avg. time $O$ |
|---|---|---|---|---|---|---|
| FC | kitchen | no | 10/10 | 5/6 | 5.8min | 6.8min |
| LC | kitchen | yes | 9/10 | 5/5 | 11.3min | 13.5min |
| FNC | liv. room, office | no | 3/3 | n/a | 10.2min | n/a |

Table 1: Runtimes for the three cases tested: full system (FC) and lesioned system (LC), both with object in canonical position; NFC: full system with object in non-canonical position. Total time to solve the task reported in minutes. The FNC case was only tested in environment $H$.

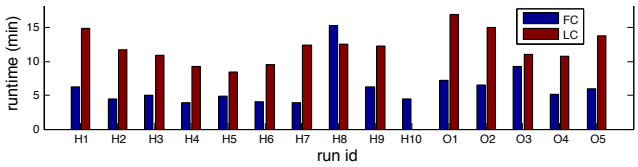



Figure 4: Total runtime in minutes of the individual runs in the FC (15 runs) and LC configuration (14 runs, no H10) for environment $H$: runs H1-H10 and $O$: runs O1-O5.

**Results** The cornflakes box was found by the robot in 32 of the 34 runs. In the two failed runs the robot maneuvered itself into a corner of the room and required human intervention. The total execution time of the successful $FC$ and $LC$ runs are plotted in Fig. 4. Hypothesis (i) claims that the robot is able to be exploit the evidence gained from perceiving its environment by integrating this with conceptual knowledge about the commonalities of such environments. That our system does this is confirmed by a significant difference (Mann-Whitney test $p < 0.01$ for both environments) in average runtime reported in Tab. 1 for these two configurations. Looking at the typical sequence of actions for the $FC$ configuration it becomes apparent that planning inferred lower costs for driving into the kitchen to begin searching (despite that being an extra distance to travel without looking for objects). This observation also explains the relative improvement of $FC$ in the larger $O$ environment, comprising larger space that has to be searched exhaustively, being comparatively higher than in $H$.

The robot was able to exploit the appearance properties for the room it started in and infer that this is unlikely to be a kitchen, as indicated by the pie charts in Fig. 3. This caused it to choose to visit the uncategorised room first in the hope that it is a kitchen. In all lesioned runs ($LC$), the robot is not able to exploit this knowledge and consequently has to conduct an exhaustive search of all rooms, optimistically starting in the room it is in, causing significantly longer runtimes.

The outlier $FC$ run number 8 shows that our system is also able to cope with a deviation in sensing, confirming hypothesis (ii). In this case the robot also first drove to the kitchen, but failed to detect the object (due to an object detection false negative). Accordingly the robot went back to the living room to continue its search there. This was due the non-zero probability of finding objects also in non-canonical locations. However, after exploring several places there, the likelihood of finding the object in the kitchen by looking again became

higher, so the robot went back and finally found it. With regard to hypothesis (iii) we can confirm that the robot was able to solve non-canonical configurations 100% of the time. In these runs, the robot also first searched in the kitchen before returning to the other room and eventually finding the object there. A system being entirely determined about the cornflakes being in kitchens, as it has been in our previous system, would not have been able to consider this alternative.

## 7 Conclusion

In this paper we presented a mobile robot system that integrates two original approaches for representing and reasoning about uncertainty. The first is a representation of space that combines knowledge about its qualitative structure (e.g. that cornflakes boxes are are type of cereal box), with probabilistic knowledge (e.g. that cereal boxes are found in kitchens 33% of the time). The second is a continual planning and execution monitoring system that employs *switching* to plan for *very* large partially observable problems that are posed by our detailed source of spatial knowledge. It is important to note that the *integration* of these approaches is crucial to the success of our work. Without the novel planner, the representation would not be capable of influencing behaviour. Without the novel representation, the planner would not be able to reason over both probabilistic instance *and* conceptual knowledge at the same time. We evaluated this combination in our robot in two real-world environments, and found that it is able to yield efficient and robust behaviours in an object search task.

## References


[Bouguerra *et al.*, 2007] A. Bouguerra, L. Karlsson, and A. Saffiotti. Handling uncertainty in semantic-knowledge based execution monitoring. In *Proc. Int. Conf. Intelligent Robots and Systems (IROS)*, pages 437–443, November 2007.

[Brunskill and Russell, 2010] E. Brunskill and S. Russell. Rapid: A reachable anytime planner for imprecisely-sensed domains. In *UAI*. AUAI Press, 2010.

[Folkesson *et al.*, 2007] J. Folkesson, P. Jensfelt, and H. I. Christensen. The m-space feature representation for SLAM. *IEEE Trans. Robotics*, 23(5):1024–1035, October 2007.

[Galindo *et al.*, 2005] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. A. Fernandez-Madrigal, and J. Gonzalez. Multi-hierarchical semantic maps for mobile robotics. In *Proc. Int. Conf. Intelligent Robots and Systems (IROS)*, pages 2278–2283, August 2005.

[Gross *et al.*, 2009] H.-M. Gross, H. J. Böhme, C. Schröter, S. Müller, A. König, E. Einhorn, C. Martin, M. Merten, and A. Bley. TOOMAS: Interactive Shopping Guide robots in everyday use-final implementation and experiences from long-term field trials. In *Proc. Int. Conf. Intelligent Robots and Systems (IROS)*, pages 2005–2012, 2009.

[Helmert, 2006] M. Helmert. The fast downward planning system. *J. Artif. Intell. Research*, 26:191–246, 2006.

[Kraft *et al.*, 2008] D. Kraft, E. Başeski, M. Popović, A. M. Batog, A. Kjær-Nielsen, N. Krüger, R. Petrick, C. Geib, N. Pugeault, M. Steedman, T. Asfour, R. Dillmann, S. Kalkan, F. Wörgötter, B. Hommel, R. Detry, and J. Piater. Exploration and planning in a three-level cognitive architecture. In *Proc. Int. Conf. Cognitive Systems (CogSys)*, 2008.

[Lauritzen and Richardson, 2002] S. L. Lauritzen and T. S. Richardson. Chain graph models and their causal interpretations. *J. Roy. Statistical Society, Series B*, 64(3):321–348, 2002.

[Maier and Steinbach, 2010] W. Maier and E. Steinbach. A Probabilistic Appearance Representation and Its Application to Surprise Detection in Cognitive Robots. *IEEE Trans. Autonomous Mental Development*, 2(99):1, 2010.

[Mooij, 2010] J. M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *J. Mach. Learn. Res.*, 11:2169–2173, August 2010.

[Mörwald *et al.*, 2010] T. Mörwald, J. Prankl, A. Richtsfeld, M. Zillich, and M. Vincze. BLORT – The Blocks World Robotic Vision Toolbox. In *Proc. ICRA Workshop Best Practice in 3D Perception and Modeling for Mobile Manipulation*, 2010.

[Pronobis *et al.*, 2010] A. Pronobis, O. M. Mozos, B. Caputo, and P. Jensfelt. Multi-modal semantic place classification. *Int. J. Robot. Res.*, 29(2-3):298–320, February 2010.

[Ross *et al.*, 2008] S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. Online planning algorithms for POMDPs. *J. Artif. Int. Res.*, 32:663–704, July 2008.

[Shani *et al.*, 2008] G. Shani, P. Poupart, R. Brafman, and S. E. Shimony. Efficient add operations for point-based algorithms. In *Proc. Int. Conf. Automated Planning and Scheduling (ICAPS)*, pages 330–337, 2008.

[Shubina and Tsotsos, 2010] K. Shubina and J. Tsotsos. Visual search for an object in a 3D environment using a mobile robot. *Computer Vision and Image Understanding*, 114(5):535–547, 2010.

[Silver and Veness, 2010] D. Silver and J. Veness. Monte-carlo planning in large POMDPs. In *Proc. Conf. Neural Information Processing Systems (NIPS)*, 2010.

[Sjöö *et al.*, 2010a] K. Sjöö, A. Aydemir, D. Schlyter, and P. Jensfelt. Topological spatial relations for active visual search. Technical Report TRITA-CSC-CV 2010:2 CVAP317, Centre for Autonomous Systems, KTH, Stockholm, July 2010.

[Sjöö *et al.*, 2010b] K. Sjöö, H. Zender, P. Jensfelt, G.-J. M. Kruijff, A. Pronobis, N. Hawes, and M. Brenner. The Explorer system. In H. I. Christensen, G.-J. M. Kruijff, and J. L. Wyatt, editors, *Cognitive Systems*, pages 395–421. Springer, April 2010.

[Talamadupula *et al.*, 2010] K. Talamadupula, J. Benton, S. Kambhampati, P. Schermerhorn, and M. Scheutz. Planning for human-robot teaming in open worlds. *ACM Trans. Intell. Syst. Technol.*, 1:14:1–14:24, December 2010.

[Vasudevan and Siegwart, 2008] S. Vasudevan and R. Siegwart. Bayesian space conceptualization and place classification for semantic maps in mobile robotics. *Robot. Auton. Syst.*, 56:522–537, June 2008.

[X, 20xx] X. Y, 20xx. anonymised for review process.

[Yoon *et al.*, 2007] S.-W. Yoon, A. Fern, and R. Givan. FF-replan: A baseline for probabilistic planning. In *Proc. Int. Conf. Automated Planning and Scheduling (ICAPS)*, 2007.

[Younes *et al.*, 2005] H. L. S. Younes, M. L. Littman, D. Weissman, and J. Asmuth. The first probabilistic track of the international planning competition. *J. Artif. Intell. Res.*, 24:851–887, 2005.

## A.2 Home Alone: Autonomous Extension and Correction of Spatial Representations

**Bibliography** Nick Hawes, Marc Hanheide, Jack Hargreaves, Ben Page, Hendrik Zender, and Patric Jensfelt. Home Alone : Autonomous Extension and Correction of Spatial Representations. In *Proc. Int. Conf. on Robotics and Automation*, 2011

**Abstract** In this paper we present an account of the problems faced by a mobile robot given an incomplete tour of an unknown environment, and introduce a collection of techniques which can generate successful behaviour even in the presence of such problems. Underlying our approach is the principle that an autonomous system must be motivated to act to gather new knowledge, and to validate and correct existing knowledge. This principle is embodied in Dora, a mobile robot which features the aforementioned techniques: shared representa- tions, non-monotonic reasoning, and goal generation and management. To demonstrate how well this collection of techniques work in real-world situations we present a comprehensive analysis of the Dora systems performance over multiple tours in an indoor environment. In this analysis Dora successfully completed 18 of 21 attempted runs, with all but 3 of these successes requiring one or more of the integrated techniques to recover from problems.

**Relation to WP** In this paper we report on further in-depth evaluation of the Dora 1 system, carried out in a real world environment of a student apartment. The focus of this evaluation was on the non-monotonic reasoning about space being part of WP3 and WP6, and the goal management framework developed as part of WP1.

# Home Alone: Autonomous Extension and Correction of Spatial Representations

Nick Hawes, Marc Hanheide, Jack Hargreaves, Ben Page, Hendrik Zender, Patric Jensfelt

***Abstract*— In this paper we present an account of the problems faced by a mobile robot given an incomplete tour of an unknown environment, and introduce a collection of techniques which can generate successful behaviour even in the presence of such problems. Underlying our approach is the principle that an autonomous system must be motivated to act to gather new knowledge, and to validate and correct existing knowledge. This principle is embodied in Dora, a mobile robot which features the aforementioned techniques: shared representations, non-monotonic reasoning, and goal generation and management. To demonstrate how well this collection of techniques work in real-world situations we present a comprehensive analysis of the Dora system's performance over multiple tours in an indoor environment. In this analysis Dora successfully completed 18 of 21 attempted runs, with all but 3 of these successes requiring one or more of the integrated techniques to recover from problems.**

## I. Introduction

Service robots working in human environments will require a lot of knowledge about their surroundings in order for them to discharge their duties successfully (e.g. maps, names for rooms etc.). In existing work this knowledge is assumed to be provided to the system during a *tour* in which a human guides the robot through the environment indicating important physical features and also providing associated descriptions. For a human, a tour is a rather natural way to convey information to a robot. However, this process is fraught with problems from a robot's point of view.

Fig. 1: Dora in the kitchen; a room of the flat the experiments were conducted in.

Nick Hawes, Marc Hanheide, Jack Hargreaves, and Ben Page are with the School of Computer Science at University of Birmingham; email: n.a.hawes@cs.bham.ac.uk. Hendrik Zender is with the German Research Center for Artificial Intelligence (DFKI), Saarbrücken. Patric Jensfelt is with the Royal Institute of Technology (KTH), Stockholm. The research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement No. 215181, CogX.

These problems include those familiar to roboticists (sensor noise, incorrect results generated by components) and those which are based more on a typical user's lack of knowledge about a robot's requirements. In this paper we present an analysis of how an intelligent mobile robot can be designed to overcome some of the problems that it will face following a home tour by a naïve user. From this analysis we developed a system with a range of behaviour and reasoning capabilities, including functionality related to the explicit representation of *gaps in its knowledge* and the ability to reason about how to act to fill these gaps. Our analysis focuses on how these abilities allow the system to recover from common errors that occur during tours through an environment for the first time. It is supported by evidence gathered from multiple runs of the robot in a real flat.

Our system features two main advances beyond previous work: *a drive to validate and refine spatial knowledge* (to detect and correct gaps and errors); and *non-monotonic reasoning about spatial knowledge* (necessary to support the revision of the robot's representations). These are presented in Section III. Sections IV and V present quantitative and qualitative analyses of our system over 18 successfully extended tours. Before this, Section II presents some background and existing approaches to dealing with typical tour-related problems.

## II. Background & Related Work

A tour as a mechanism for providing a robot with information is a common meme in autonomous and interactive robotics (e.g. [1], [2]). Whilst previous work has identified problems that occur during tours due to the guide's incorrect expectations about a robot's abilities [3], to date no work has provided an account of the problems that occur from a robot's point of view, and what mechanisms can cope with them. When referring to "problems" we don't mean the ability for a robot to solve a particular fixed problem. Instead we refer to cases where either the robot or the human have (in some sense) functioned incorrectly.

From the robot's point of view we can distinguish *two general types of problems*: those caused by the robot not functioning correctly in a designer-anticipated context (e.g. the failure of a perceptual routine resulting in an object or feature going unobserved), and those caused by the robot's environment changing to become out of sync with the robot's current representation of it (e.g. a human closing a door which was open during the tour). Given the limited knowledge of current robots (including our own), these different types of problems are actually indistinguishable at a system level. Both result in the robot having a representation of the environment which does not reflect the true state of the world. Thus we propose that both problem types can be addressed by ensuring that robots feature a motivating drive *to ensure that their representations are as complete and as correct as possible*. Given that a tour only provides a robot with part of the knowledge it needs, any service robot in this situation must therefore be able to *autonomously explore* its environment (after the tour) in order to fill any knowledge gaps (following the principle of *discovery* [4]). And given that the aforementioned problems can occur both during the tour and during autonomous exploration, it is also important that a robot never assumes that its knowledge is ever entirely complete or correct. Instead the aforementioned drive should give rise to behaviours which attempt to *validate and refine* its knowledge as appropriate, in addition to extending it. This behaviour is an essential part of any system that must simultaneously learn about, and perform tasks in, the real world. Such systems do not have the luxury of separate training and test phases, and instead must learn online whilst acting.

It could be argued that all the problems we expect a robot to encounter are actually the result of a badly designed or engineered system, or a poorly trained user. However, given the current state-of-the-art it is unrealistic to expect that complex intelligent robots deployed into human-populated environments will not encounter these problems, or similar ones. The wide range of contexts these systems must cope with, and the behaviours they must be capable of generating, mean that it may prove just too difficult to fully debug a whole system before it is deployed in its target environment. Thus a robot must be equipped with systems that mitigate the effects of run-time problems. It is also unrealistic to expect any non-expert user to be able to reliably provide a current robot system with all of the information it actually needs during a tour. For example, localisation and navigation systems typically require detailed maps of an environment in order to function correctly, and it is clear that a busy user may only show a robot some subset of the environment before moving on to another task.

Existing work typically relies on one of two different approaches to coping with problems at run-time: *autonomy* or *interaction*. Autonomous approaches to coping with localisation errors typically leverage the probabilistic machinery they are implemented with. For example, most existing systems are capable of characterising their mapping and localisation uncertainty, some are capable of explicitly acting to reduce this uncertainty [5], and others can explicitly represent possibilities for extending their metric spatial knowledge beyond the frontiers of their maps [6]. Interactive approaches require that the robot asks the human to provide input if it encounters a problem. For example, systems have been developed which ask a human whether it has just passed a door [2], and which use uncertain categorisations as the basis for dialogue about room categories [1] and object features [7].

In this paper we focus on the autonomous resolution of some prototypical instances of perceptual problems: the possible failure of a door detector to identify doors in the robot's environment, and problems in a spatial model resulting in either failure to identify unvisited areas of space or failure to move the robot correctly.

## III. System Architecture

We have developed an architecture for an intelligent mobile robot which embodies the principles described in the previous section. This architecture is deployed as part of *Dora*, a robot developed in the CogX project[1] (see Fig. 1). Dora has been designed to be able to autonomously explore an unknown environment and patrol a known one. It is also able to autonomously determine the functional category of rooms (to support future human-robot interaction and service tasks). Dora explicitly represents gaps in its spatial model (i.e. its map and its knowledge of room categories) and is able to take action to fill these. When displaying nominal behaviour, this model is in alignment with its environment; it contains representations of all visited rooms and their categories. In case of misalignment resulting from problems, Dora has mechanisms for updating these representations.

The Dora architecture is an instantiation of the CoSy Architecture Schema, and is implemented using the associated software toolkit, CAST [8].

[1] `http://cogx.eu`

This schema is based on a shared working memory model and is designed to support flexible, parallel, information sharing in heterogenous, yet integrated, systems. This support is an essential requirement for robots such as Dora which must share a lot of explicit knowledge between its subsystems and where this knowledge must be both used and refined concurrently.

In the following subsections we present two of the subsystems which allow Dora to act to make its representation as complete and correct as possible when faced with the complexity and dynamics of the real world: its multi-layered spatial model and its goal management framework. These subsystems are jointly revising representations and mutually notify each other about the modifications they make through the CAST framework. For a more detailed account of the rest of the Dora architecture, see previous work, e.g. [9], [10].

### A. Reasoning with changing spatial knowledge

As a mobile robot, Dora requires a map-based spatial representation for localisation and navigation. We take the multi-layer hybrid approach to map representation described in [10]. Particularly relevant for this paper are the map's *place* and *conceptual* layers. Fig. 2 shows visualisations of maps generated by Dora. The colouring of the place nodes represents the information about the containment of places within rooms as provided by the conceptual layer.

The place layer is responsible for discretising the continuous representation of space generated by the layers below it (using a metric SLAM approach) into a graph-based representation of free space. The nodes in the graph are called places, and they are created at $1m$ intervals along the robot's trajectory. Graph-edges indicate immediate adjacency of places and the possibility of moving between them. Gateways play an important role for clustering places into larger coherent areas. In buildings, doorways are typical gateways. A *door detection* component inspects laser scans to find width changes that look like doorways and triggers the place layer to mark the corresponding place as a *gateway place*. The place layer can also generate *placeholders*. A placeholder represents an unexplored direction that the robot might move in, thus explicitly representing the possibility of creating additional places beyond the limits of the current place graph. The place layer is connected to a navigation module that can move the robot to places and placeholders.

It is important that autonomous robots which are supposed to operate in domestic environments have a notion of spatial units that are also meaningful for humans. In our system, the conceptual layer (an extended and improved variant of the conceptual map described in [11]), is responsible for linking the other layers of the spatial model (including the place layer) to concepts which humans can relate to. The conceptual layer represents map knowledge in an OWL-DL ontology consisting of a taxonomy of spatial concepts (*TBox*), and how they can be characterised in terms of human-compatible categories, as well as the knowledge about individuals in the domain (*ABox*). It also makes use of a combined rule and OWL-DL reasoner based on the Jena Framework to perform different reasoning tasks on the ontology. The taxonomy defines spatial regions called *areas* as the basic units corresponding to a human-compatible segmentation of space. We distinguish between two basic kinds of areas. *Rooms* are spatial areas whose primary purpose is defined by the kinds of actions they afford (e.g., WordNet defines a kitchen as "a room equipped for preparing meals"). Here, we make the simplified assumption that the presence of certain objects (e.g. kettle, toothpaste) determines the respective subcategory of a room (e.g. kitchen, bathroom). *Passages* (e.g. corridors) are areas whose primary purpose is to link rooms and provide access to other areas.

A prerequisite for reasoning about room categories is to have a notion of rooms. Based on the information about the connectivity of places and whether they constitute gateways or not, the conceptual layer forms areas (rooms or corridors) by clustering places that are transitively interconnected without passing a doorway. Since both door and object detection can malfunction, room formation and categorisation must be *non-monotonic* processes in order to support the potential for knowledge revision. Room formation and maintenance is handled by a general purpose rule engine, while the OWL-DL reasoner is used to infer which categories can be applied to known rooms. Both are able to make non-monotonic inferences in the ABox: whenever a previously true condition turns false, the conclusions drawn from it are retracted. This means that the conceptual layer is capable of correcting both the assignment of places to rooms and the assignment of categories to rooms when additional (counter-)evidence becomes available.

In order to perform room categorisation, Dora must populate the conceptual layer's ABox with knowledge about objects in the environment. To

do this Dora has an *active visual search* behaviour. This moves Dora around the current room running an object detector using a set of pre-trained models. Our visual search implementation is a derivation of the randomised art gallery algorithm which only looks at points in the room which are likely to contain objects [12]. These points are currently linked to obstacles in Dora's metric map.

### B. Behaviour generation

Dora's behaviour is generated and coordinated by a continual planning and execution system which is controlled by a goal generation and management framework. This framework is responsible for generating new goals (i.e. descriptions of desired future states) for Dora from the outputs of sensors and other processing, selecting which goals should be followed, triggering the planner to create a plan to achieve the selected goals, then managing the execution of subsequent plans. Dora has *goal generators* which aim to satisfy its overall drive to have accurate knowledge about its environment (as discussed in Section II) by creating individual goals to *explore* each generated placeholder (yielding a more complete place layer) and to *categorise* each generated room (yielding a more complete conceptual layer). In addition to this, Dora also generates a goal to *patrol* each previously generated place (and consequently each area too). This allows it to revisit known space, thus validating previously generated knowledge. If a goal is no longer valid (if it has been achieved or it is not longer appropriate) the goal generator which created it removes it from the framework.

Dora's goal management framework is responsible for selecting which of the generated goals should be forwarded to planning and execution. A goal which has been thus selected is referred to as *activated*. Before activation it must pass through two additional stages. After generation, goals are first *unsurfaced*. They must then pass through a bank of *filters* to become *surfaced* before they can be considered for activation. The management processes move goals between these stages based on a variety of conditions. In Dora, goals are surfaced based on their type (all the goal types described above are automatically surfaced) and only unsurfaced if the system fails to achieve them more than a set number of times (currently 10).

Goals are activated based on a combination of features. The current implementation ranks all goals based on the number of previous attempts to achieve them, then by priority, then by a calculation of information gain versus estimated cost. Only the top ranked goal is then selected for activation. In Dora the information gain for exploring a placeholder is related to the amount of free space it covers, the information gain for categorising a room is related to the number of places it contains, and for a patrolling a place it is related to the last time the place was visited. Cost estimates are based on the distance Dora would have to travel to achieve the goal. These values are associated with a goal by its generator. The generator is also responsible for maintaining these values as state changes occur (e.g. the robot moves), and also for removing goals that are no longer justified by the robot's knowledge (e.g. when a placeholder is visited or a room is categorised). A goal is assigned a priority when it is generated. Currently these priorities are inherited from priorities manually assigned to their generators. Our current scheme assigns explore and categorise goals the same priority, with patrol goals having a lower priority. This reflects the need to generate knowledge before it can be validated or revised. The effects of this decision are seen in the case studies discussed below. Ranking goals first by the number of previous attempts to achieve them allows goals to be *postponed* when Dora fails to achieve them (as they are effectively relegated to the end of the ranked activation list). This is designed to prevent the system from repeatedly trying and failing to achieve one goal when others still exist. Currently Dora detects some failures by waiting for the execution system to timeout. The effects of this design are also discussed below.

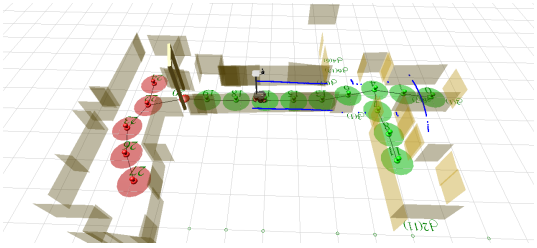
The use of a continual planner is essential for a robot in a dynamic environment. Changes in the environment and sensor noise mean that plans often need to be revised on-the-fly as they either become impossible or more information becomes available, allowing other plans to become possible. The continual planner we use is specifically designed to detect such changes during plan execution, and replan as appropriate [13]. This allows Dora to cope with open world problems (e.g. not knowing how many rooms exist in advance) in a deliberative way.

## IV. Experiments

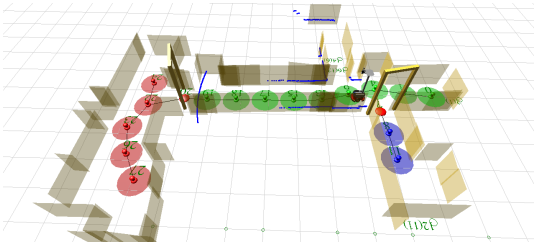
To evaluate how the Dora system copes with the problems that occur in realistic settings we took the robot out of the lab and into a flat in Birmingham, UK[2]. Here we ran Dora multiple times and gathered data about its performance[3]. Dora (displayed

[2]Mason Hall, `http://j.mp/9tLEUx`

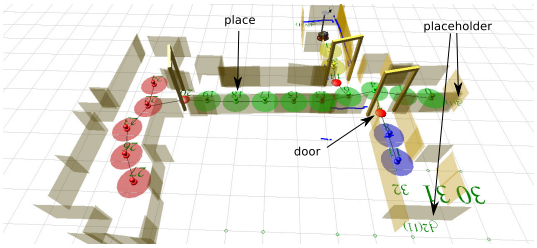
[3]Video available at `http://cogx.eu/results/dora`



(a) The door to the lower-right room has not been detected.



(b) Door detection triggers a revision of the representation.




(c) All rooms are now found.

Fig. 2: Snapshots from the evolution of the spatial representation during one run of Dora. Placeholder are depicted as small unfilled green circles, places are solid disks. This representation is entirely created from scratch each run as part of the tour and the autonomous exploration.

in Fig. 1) is based on a MobileRobots P3DX platform equipped with a Hokuyo laser scanner and a pan-tilt unit holding two Flea2 cameras. The software system runs on a single dual-core laptop. This section describes the methodology we followed when running the system and presents some descriptive statistics generated from this experiment.

### A. Procedure

The environment we operated Dora in was composed of a kitchen, a corridor, and two bedrooms. Dora was initially given a short tour, then required to build a complete map of this environment and categorise all the rooms except the corridor. Each bedroom contained 3 or 4 objects that Dora could recognise and use for room categorisation. The kitchen, being larger, contained 6 objects. The objects were placed in relatively normal positions where they were visible to the robot. Fig. 2(c) shows a map of the environment as acquired by Dora. The room to the left is the kitchen, which is connected by the corridor to the bedrooms. Dora was always started at the right end of the corridor, then given a tour along the corridor, into the first bedroom (lower right in Fig. 2(c)), and then into the kitchen (ignoring the other bedroom), and no room was assigned a category. After the tour, the filter bank in the goal management framework was reconfigured to let goals surface, switching Dora to autonomous behaviour.

TABLE I: Descriptive statistics from all 18 successful runs. Standard deviation indicated by $\pm$.

| Type | (re-)activ. | duration | dur. per run |
|---|---|---|---|
| categorize room | 4.1 (1.1) | $216 \pm 117$s | 901s (63%) |
| explore place | 16.6 (0.94) | $22.5 \pm 15.6$s | 375s (27%) |
| patrol places | 4.4 (0.0) | $9.3 \pm 6.0$s | 41.4s (3%) |

We conducted a systemic analysis of all the runs, adopting the SInA methodology originally developed for human-robot interaction [14]. To support this, Dora was instrumented with logging mechanisms to record a range of state changes. The acquired logs were used to generate annotations to support qualitative analysis using visual tools [15], and quantitative analysis via descriptive statistics [16]. In the following we take a closer look at the actual behaviour represented by the creation and activation of goals in the system.

### B. Descriptive statistics

Dora met our criteria for success (i.e. explored and categorised all rooms) in 18 out of 21 attempted runs (85.7%). In the remaining three runs Dora failed to explore the bedroom which was omitted during the tour. This was because no placeholder was generated in it at any point during these runs. On average, Dora took $26.25 \pm 8.64$ (std. dev.) minutes to accomplish the full task in the 18 successful runs; of which the system spent 11.45 seconds planning for goals selected by the goal management scheme. An average of $23.52 \pm 8.93$ minutes was dedicated to autonomous behaviour, the remainder to the tour. The average total distance Dora travelled was 104 metres.

Tab. I presents some per-run statistics on active goals (those that are planned for and executed by the robot, thus generating behaviour). The "(re-)activation" column shows the average number of activations of goals of each type. In all runs, there were 3 rooms to categorise. The average number of goal activations for room categorisation was greater than three due to goal postponements. This also explains why we observe 1.1 re-activations in the case of categorise-room goals. The action execution timeout was set to 6 minutes (360 seconds) which was occasionally

not long enough for a categorisation process to complete successfully.

Tab. I also shows that 63% of the autonomous part of each run was spent categorising rooms, i.e., looking for objects. The exploration of new places accounted for 27% and only 3% were dedicated to patrolling previously-visited places. This is explained by the setup of the experiment: a run was deemed be finished when all rooms were explored and categorised. Patrol goals were assigned a lower priority than exploration and categorisation, so they were only activated in five runs when there were no other surfaced goals, but not all rooms had already been found and categorised. The remaining time in each run was taken by Dora deciding what to do next (activity planning accounted for roughly 14% of the remaining time) and general communication overhead between components.

## V. Case Studies

The following sections present a detailed analysis of the different types of system runs we observed during our experimentation. We start with the case in which no problems occur and follow this with cases that cover a number of problems that occurred during the experiments.

### *Case A: Ideal case*

In the ideal case Dora successfully detects all doors when first passing through them, puts placeholders in the bedroom which was not entered during the tour (thus allowing it to autonomously explore it), and achieves all goals on the first attempt. It should be noted that the ideal case requires the majority of Dora's capabilities, including autonomous extension and handling of incomplete knowledge. It doesn't require any non-monotonic reasoning or active validation of acquired knowledge because the knowledge is correct when initially obtained.

Fig. 3 shows the progress of a run of Dora. The run illustrated in the figure is *not* an ideal one in total, it is included to serve all the case studies. However, if we look only at the first 10 minutes (600 seconds) it presents the prototypical behaviour of Dora. In this behaviour placeholders, uncategorised rooms, and existing places give rise to goals to explore, categorise, or patrol them, respectively. In the ideal case patrol goals never get activated, as the experiment is finished once all placeholders have been explored or discarded and all rooms have been categorised. Patrol goals are assigned a lower priority, so they are only pursued if no other goals exist. In the figure we can see the two phases of the experiment. During the tour, all the goals were unsurfaced (represented as green bars). But it can be seen that during the tour (unsurfaced) goals were created whenever a placeholder was assigned to an area of open space that had not yet been visited, or when a new (and thus uncategorised) room entity was created due to the detection of a door. During the tour, Dora frequently created placeholders directly in front of itself which were immediately visited as part of the tour and therefore removed immediately. Explore goals 3, 9, 12-16, 21, and 23 were such cases. Also, categorisation goals for the first bedroom and the kitchen (corresponding to goals "CategorizeRoom_1" and "CategorizeRoom_2") were generated during the tour because all doors were correctly detected. Fig. 2(b) illustrates the spatial representation of an equivalent case, where the two rooms were detected.

After roughly 130 seconds of touring the autonomous behaviour was activated. The goal management framework took control, choosing which goal to pursue next by trading predicted information gain for costs. Goals were activated and then corresponding plans executed. Dora's behaviour caused new goals to be created. In Fig. 3, "Explore_25" was the first goal to be created while the robot was pursuing another goal. Shortly after creation it was activated and the associated plan executed. In this case, the exploration goal was removed during plan execution because the spatial layer concluded that the placeholder was too close to an already existing place, so the creation of a new place was not possible ("activated and merged with an existing place").

The "active" line in Fig. 3 highlights the class of the currently activated goal. It shows that Dora started by exploring 4 placeholders before deciding to categorise room 2 (the kitchen). This decision was justified by the cost and gain associated with that plan at that point: Dora was in room 2 already (low cost) and it has a large area (high gain).

In the ideal case similar behaviour is demonstrated throughout the run, with Dora switching between successful exploration and categorisation until the task is complete. However, due to the nature of the real world and robot perception, this is not guaranteed. Hence, we now look at cases which deviate from the ideal behaviour.

### *Case B: Missed placeholder*

The first problem case we look at is one in which Dora does not immediately discover the open space associated with the bedroom ignored

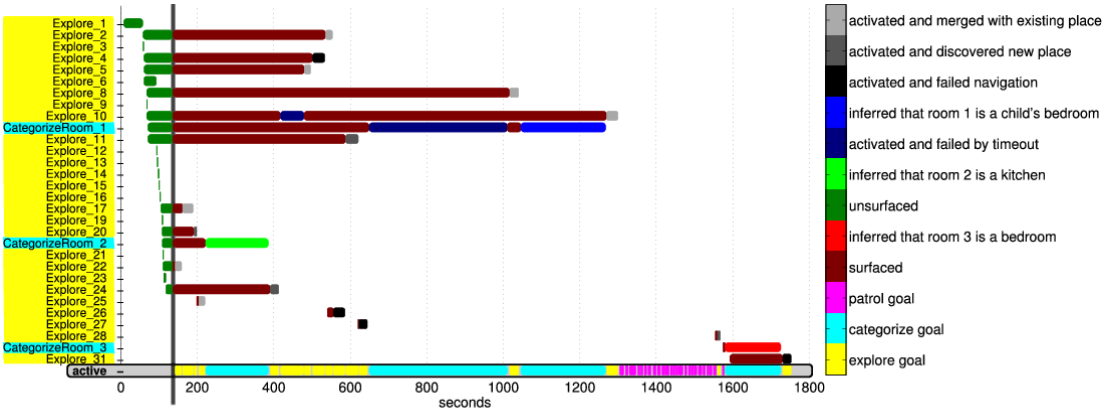



Fig. 3: A visualisation of the progress of a single Dora run. The x-axis shows the time from the start of the run. The y-axis lists all the goals generated in the run. The line titled "active" shows the type of the currently activated goal (in yellow, cyan, and magenta). The other lines show the life-time of each goal, colour-coded according to the respective status (unsurfaced, surfaced or active). Individual patrol goals are omitted for brevity. The vertical line at second 130 marks the end of the tour when the robot starts autonomous exploration.

during the tour. This occurs when no placeholder is generated inside the room and consequently no associated goal to explore is created either. Without the drive to continuously update and validate its representations, Dora would not be able to recover from this problem by revising and extending its knowledge.

This particular problem occurred in 5 of the 18 successful runs. One instance is depicted in Fig. 3. We can see that by 1300s only two rooms had been discovered and there are neither explore nor categorise goals left to pursue. At this point, Dora believes she has explored and categorised everything. She still has the drive to validate her representation so she starts patrolling, motivated by the lower priority patrol goals. Though these goals are not individually visualised we can see a series of (pink) patrol goal activations in the "active" line. During this period Dora patrolled from place to place, eventually reaching the place in front of the bedroom door. This time, the open space *was* detected and a placeholder generated, triggering the creation of the "Explore_28" goal. Because explore goals are assigned a higher priority than patrol goals, Dora immediately stopped patrolling and started exploring into that open space. While exploring, the robot detected the door and concluded that this is a new, yet unknown, room. A corresponding categorisation goal ("CategorizeRoom_3") was created and subsequently activated. After finding two objects, Dora concluded that this room was in fact a bedroom. It can be noted that during the active visual search in that room the robot also detected more open space (cf. the "Explore_31" goal). So, the system is capable of handling side effects of actions. To end this case, Dora then also explored this last placeholder, finally accomplishing the overall task successfully.

### *Case C: Undetected door*

Dora is designed to detect doors when passing through them. Detection is performed by inspecting the laser scan for changes in width which might indicate the narrower passage of a door frame. This process is not completely reliable due to sensor noise and changes in robot orientation when moving. This means occasionally (on 1 from the 18 successful runs) Dora does not detect a door when one is present in the environment. Fig. 2(a) visualises how this occurred during our experiments: the door between the corridor and the first bedroom (lower right) was not detected during the tour, so the bedroom was initially considered to be an extension of the corridor. Later, when Dora was exploring a placeholder close to the door, she finally detected it (see Fig. 2(b)). This caused the conceptual layer to revise its representation, producing two rooms from the previously existing one (and yielding the correct map shown in Fig. 2(c)). This case underlines the necessity for non-monotonic reasoning and dynamic goal generation. It was only after revising its representation in the light of new information that Dora could determine that although it had already visited the newly distinguished room, it still lacked a categorisation result for it.

### *Case D: Timeouts and re-activations*

A problem that occurs for the Dora system once the initial tour is over is that a planned action occasionally results in actual robot behaviour that both fails to generate the intended effects, and fails to

terminate. For example, a navigation command to move the robot to a place or placeholder may rarely encounter a problem which gets Dora stuck somewhere, and occasionally the active visual search will fail to find any objects at all. This problem is due to the planning actions being an opaque abstraction over the implementing techniques. This abstraction prevents components performing detailed reasoning about the conditions in which the actions can safely be executed, and the full range of their possible effects. In other words the world model Dora uses for reasoning does not capture the full range of (uncertain) knowledge it should. We address this problem by taking the, admittedly simplistic, approach of defining timeouts for the achievement of each type of goal (as mentioned in Sec. III-B). For instance, we set the timeout for achieving room categorisation goals to 360 seconds. The aim of this approach is to allow Dora to postpone goals that can't be achieved due to problems caused by the aforementioned incomplete world model. In our experiments, there was more than one such occasions per run in average, with only 5 runs featuring no such case.

An example of this behaviour is apparent in Fig. 3. The goal "CategorizeRoom_1" was active from 650s–1010s, but Dora failed to categorise the room. Instead of indefinitely pursuing that single goal, Dora postponed it and instead continued exploring, re-activating the still surfaced categorisation goal later on. The "Explore_10" goal from the same run shows a similar life cycle.

## VI. Conclusion

We started this paper by describing our general view of the problems faced by a robot when placed in an unknown environment. From this view we derived a principle that such a robot should be driven to ensure that its knowledge of its environment is complete and correct. This led us to conclude that robots must be motivated to extend, validate and refine their knowledge in order to provide a representation to support action. We then described Dora, an intelligent mobile robot which instantiates these principles using *shared representations*, *non-monotonic reasoning*, and a *goal generation and management framework*, with a particular focus on spatial knowledge. To support our claims we presented a comprehensive analysis of Dora's performance when given incomplete tours of environment by a human, and then left to act autonomously. The contents of Dora's architecture allowed it to successfully complete 18 out of 21 of these runs, even in the presence of errors and unreliable perception. Only 3 of the 18 successes were ideal cases, with the remaining 15 requiring either goal management or non-monotonic reasoning to recover from one or more problems.

The analysis of the data produced by our experiments has yielded further insights into the limitations and constraints of our approach. In future work we will use these insights to support the introduction of non-monotonic processing into other Dora subsystems. We will also work towards closing the loop with humans (both during and after the tour) in order to exploit them as an additional source of knowledge.

## References


[1] J. Peltason, F. H. K. Siepmann, T. P. Spexard, B. Wrede, M. Hanheide, and E. A. Topp, "Mixed-initiative in human augmented mapping," in *Proc. Int. Conf. on Robotics and Automation*, May 2009, pp. 2146–2153.

[2] G.-J. Kruijff, H. Zender, P. Jensfelt, and H. I. Christensen, "Clarification dialogues in human-augmented mapping," in *Proc. Conf. on Human-Robot Interaction*, March 2006, pp. 282–288.

[3] K. Fischer and M. Lohse, "Shaping naive users' models of robots' situation awareness," in *Proc. Int. Symp. on Robot and Human Interactive Communication*, August 2007, pp. 534–539.

[4] D. Maio and S. Rizzi, "Clustering by discovery on maps," *Pattern Recognition Letters*, vol. 13, no. 2, pp. 89–94, 1992.

[5] F. Amigoni and V. Caglioti, "An information-based exploration strategy for environment mapping with mobile robots," *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 684–699, 2010.

[6] B. Yamauchi, "Frontier-based exploration using multiple robots," in *Proc. Int. Conf. on Autonomous Agents*, May 1998, pp. 47–53.

[7] N. Hawes, J. Wyatt, M. Sridharan, M. Kopicki, S. Hongeng, I. Calvert, A. Sloman, G.-J. Kruijff, H. Jacobsson, M. Brenner, D. Skočaj, A. Vrečko, N. Majer, and M. Zillich, "The PlayMate system," in *Cognitive Systems*. Springer, 2010, pp. 367–393.

[8] N. Hawes and J. Wyatt, "Engineering intelligent information-processing systems with CAST," *Advanced Engineering Informatics*, vol. 24, no. 1, pp. 27–39, January 2010.

[9] M. Hanheide, N. Hawes, J. Wyatt, M. Göbelbecker, M. Brenner, K. Sjöö, A. Aydemir, P. Jensfelt, H. Zender, and G.-J. Kruijff, "A framework for goal generation and management," in *Proc. AAAI Workshop on Goal-Directed Autonomy*, July 2010.

[10] J. Wyatt, A. Aydemir, M. Brenner, M. Hanheide, N. Hawes, P. Jensfelt, M. Kristan, G.-J. Kruijff, P. Lison, A. Pronobis, K. Sjöö, D. Skočaj, A. Vrečko, H. Zender, and M. Zillich, "Self-understanding and self-extension: A systems and representational approach," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 4, pp. 282–303, December 2010.

[11] H. Zender, P. Jensfelt, O. M. Mozos, G.-J. Kruijff, and W. Burgard, "Conceptual spatial representations for indoor mobile robots," *Robotics and Autonomous Systems*, vol. 56, no. 6, June 2008.

[12] A. Aydemir, K. Sjöö, and P. Jensfelt, "Object search on a mobile robot using relational spatial information," in *Proc. Int. Conf. on Intelligent Autonomous Systems*, August 2010, pp. 111–120.

[13] M. Brenner and B. Nebel, "Continual planning and acting in dynamic multiagent environments," *Journal of Autonomous Agents and Multiagent Systems*, vol. 19, no. 3, pp. 297–331, 2009.

[14] M. Lohse, M. Hanheide, K. Rohlfing, and G. Sagerer, "Systemic interaction analysis (SInA) in HRI," in *Proc. Int. Conf. on Human-Robot Interaction*, March 2009, pp. 93–100.

[15] P. Wittenburg, H. Brugman, A. Russel, A. Klassmann, and H. Sloetjes, "Elan: a professional framework for multimodality research," in *Proc. Language Resources and Evaluation Conference (LREC)*, May 2006.

[16] M. Hanheide, M. Lohse, and A. Dierker, "SALEM - Statistical AnaLysis of Elan files in Matlab," in *Proc. LREC Workshop on Multimodal Corpora: Advances in Capturing, Coding and Analyzing Multimodality*, May 2010.

## A.3 A Unifying Model-Centric Analysis Approach for Robotic Systems

**Bibliography** Jens Otto, Sebastian Wrede, Marc Hanheide, and Nick Hawes. A Unifying Model-Centric Analysis Approach for Robotic Systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011. submitted

**Abstract** Software engineering in robotics increasingly adopts a model-driven approach for the development of complex robotics systems. Despite this, popluar robotics software frameworks lack features for modeling architectural aspects, for instance the structural decomposition of the system architecture and its inter-process communication. The lack of modeling support in these frameworks makes it hard to design, verify and analyze robotics systems from an architectural viewpoint. To overcome this situation, we propose a domain-specific meta-model and a corresponding analysis toolchain which extracts models of robotics systems at runtime and is applicable to different robotics frameworks. We show the suitability of this approach by applying it to two service robots integrated with different robotics middleware frameworks and highlight possibilities for further analysis on the basis of the extracted models. Concluding, we discuss opportunities facilitated by a model-centric development process for robotics systems.

**Relation to WP** This paper reports an approach to model-centric analysis of existing system architectures. As part of the efforts in WP1 and WP7 to study architectures and assess the implemented system, a novel architecture description framework with a dedicated meta-model for CogX's integration framework CAST has been proposed and employed in the Dora 1 system.

## A.4 A system for interactive learning in dialogue with a tutor

**Bibliography** Danijel Skočaj, Matej Kristan, Alen Vrečko, Marko Mahnič, Miroslav Janíček, Geert-Jan M. Kruijff, Marc Hanheide, Nick Hawes, Thomas Keller, Michael Zillich, and Kai Zhou. A system for interactive learning in dialogue with a tutor. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011. submitted

**Abstract** In this paper we present representations and mechanisms that facilitate continuous learning of visual concepts in dialogue with a tutor and show the implemented robot system. We present how beliefs about the world are created by processing visual and linguistic information and show how they are used for planning system behaviour with the aim at satisfying its internal drive – to extend its knowledge. The system facilitates different kinds of learning initiated by the human tutor or by the system itself. We demonstrate these principles in the case of learning about object colours and basic shapes.

**Relation to WP** In this paper we present George Year 2 system. We first briefly present the individual competencies: vision (WP2), visual learning (WP5), situated dialogue (WP6), motivation (WP1) and planning (WP4), and then describe how they are integrated in the robot system (WP7).

## A.5 Learning to predict how rigid objects behave under simple manipulation

**Bibliography** Marek Kopicki, Sebastian Zurek, Rustam Stolkin, Thomas Mörwald, and Jeremy Wyatt. Learning to predict how rigid objects behave under simple manipulation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA11)*, May 2011

**Abstract** An important problem in robotic manipulation is the ability to predict how objects behave under manipulative actions. This ability is necessary to allow planning of object manipulations. Physics simulators can be used to do this, but they model many kinds of object interaction poorly. An alternative is to learn a motion model for objects by interacting with them. In this paper we address the problem of learning to predict the interactions of rigid bodies in a probabilistic framework, and demonstrate the results in the domain of robotic push manipulation. A robot arm applies random pushes to various objects and observes the resulting motion with a vision system. The relationship between push actions and object motions is learned, and enables the robot to predict the motions that will result from new pushes. The learning does not make explicit use of physics knowledge, or any pre-coded physical constraints, nor is it even restricted to domains which obey any particular rules of physics. We use regression to learn efficiently how to predict the gross motion of a particular object. We further show how different density functions can encode different kinds of information about the behaviour of interacting objects. By combining these as a product of densities, we show how learned predictors can cope with a degree of generalisation to previously unencountered object shapes, subjected to previously unencountered push directions. Performance is evaluated through a combination of virtual experiments in a physics simulator, and real experiments with a 5-axis arm equipped with a simple, rigid finger.

**Relation to WP** This paper presents approaches to learning to predict behaviour of objects during real word robot-environment interaction. The presented work contributes to the robot system WP2 (object perception and manipulation).

# Learning to predict how rigid objects behave under simple manipulation

Marek Kopicki[1], Sebastian Zurek[1], Rustam Stolkin[1], Thomas Mörwald[2], Jeremy Wyatt[1]
[1]School of Computer Science, University of Birmingham, UK
[2]Automation and Control Institute, Vienna University of Technology, AT

***Abstract*— An important problem in robotic manipulation is the ability to predict how objects behave under manipulative actions. This ability is necessary to allow planning of object manipulations. Physics simulators can be used to do this, but they model many kinds of object interaction poorly. An alternative is to learn a motion model for objects by interacting with them. In this paper we address the problem of learning to predict the interactions of rigid bodies in a probabilistic framework, and demonstrate the results in the domain of robotic push manipulation. A robot arm applies random pushes to various objects and observes the resulting motion with a vision system. The relationship between push actions and object motions is learned, and enables the robot to predict the motions that will result from new pushes. The learning does not make explicit use of physics knowledge, or any pre-coded physical constraints, nor is it even restricted to domains which obey any particular rules of physics. We use regression to learn efficiently how to predict the gross motion of a particular object. We further show how different density functions can encode different kinds of information about the behaviour of interacting objects. By combining these as a product of densities, we show how learned predictors can cope with a degree of generalisation to previously unencountered object shapes, subjected to previously unencountered push directions. Performance is evaluated through a combination of virtual experiments in a physics simulator, and real experiments with a 5-axis arm equipped with a simple, rigid finger.**

## I. Introduction

This paper presents algorithms which learn to predict the motion of a rigid object resulting from an robot push. These algorithms do not rely on any encoding of Newtonian mechanics, but can be trained online. Object interactions are learned as distributions. Our system does not know a priori about impenetrability, gravity, or kinematic relations between objects, all being learned from data.

Although work has been done on push manipulation in robots [1], [2], [3], [4] it is restricted to planar sliding motions of what are effectively 2D objects. There is little literature addressing the more complex problem of push manipulations on real 3D bodies, which are free to tip or roll. It is possible to use physics simulators to predict the motion of interacting rigid bodies. However, this approach is reliant on explicit knowledge of the objects, the environment, and key physical parameters which can be difficult to tune. Even then, such predictions may not be possible due to inherent limitations of the physical model employed, for example when modeling friction.

Machine learning approaches have been developed to learn to classify or provide predictions for objects or object classes, e.g. rolling versus non-rolling objects [5], [6], or liftable versus non-liftable objects [7]. These kinds of approach are limited, in that predictions learned may not be generalisable to a new object, pose or push direction, and explicit 6-DOF rigid body motions are not predicted. In contrast, our approach learns to make predictions of explicit 3D rigid body transformations. The probabilistic nature of the learning enables generalisation to novel push directions, object poses, and objects with novel shapes.

This paper extends our previous work [8] in three ways. First, we modify the prediction scheme to make use of local coordinate systems that move with parts of the object. This improves learning and generalisation, since now we predict relative rather than absolute changes in pose. Second, we show how a two expert approach can be extended to include a combination of many experts, which encode new information about how objects interact. This change allows generalisation with respect to both push direction, and object shape. Third, we implement a version of our prediction scheme based on regression, and show how it can efficiently learn the gross motion characteristics of a particular object, although it can struggle with certain kinds of generalisation. Finally we present results from physical experiments in which various real objects were subjected to complex 3D motions, such as tipping and toppling, while pushed by a real robot. The real experiments are additionally supported by an extensive set of simulation experiments.

## II. Representations

Consider three reference frames $A$, $B$ and $O$ in a 3-dimensional Cartesian space (see Figure 1). While frame $O$ is fixed, $A$ and $B$ change in time and are observed at discrete time steps $..., t-1, t, t+1, ...$ every non-zero $\Delta t$. A frame $X$ at time step $t$ is denoted by $X^t$, a rigid body transformation between a frame $X$ and a frame $Y$ is denoted by $T^{X,Y}$.

From classical mechanics we know that in order to predict a state of a body, it is sufficient to know its mass, velocity and a net force applied to the body. We do not assume any knowledge of the mass and applied forces, however the transformations of a body, with attached frame $B$, over two time steps $T^{B_{t-1},B_t}$ and $T^{B_t,B_{t+1}}$ encode its acceleration - the effect of the applied net force. Therefore, if the net force and the body mass are constant, the transformations $T^{B_{t-1},B_t}$ and $T^{B_t,B_{t+1}}$ provide a complete description of

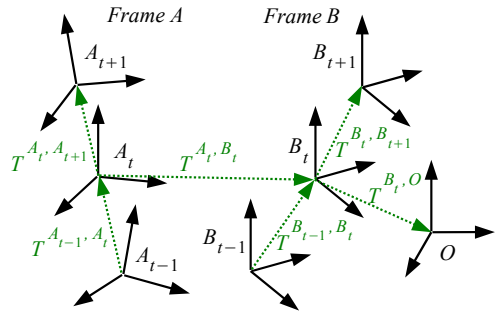



Fig. 1. A system consisting of two interacting bodies with frames $A$ and $B$ in some constant environment with frame $O$ can be described by six rigid body transformations $T^{A_t,B_t}$, $T^{B_t,O}$, $T^{A_{t-1},A_t}$, $T^{A_t,A_{t+1}}$, $T^{B_{t-1},B_t}$, and $T^{B_t,B_{t+1}}$.

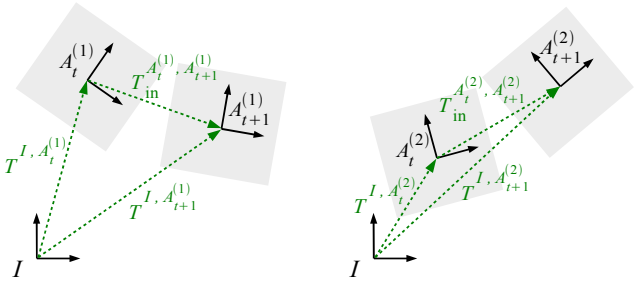



Fig. 2. In the above two scenes a pose change between time step $t$ and $t+1$ as observed in instantaneous object body frame $A^{(1)}$ and the same object in another instantaneous body frame $A^{(2)}$ given inertial frame $I$ are both the same. However because transformations $T^{I,A^{(1)}}$ and $T^{I,A^{(2)}}$ are different, the corresponding transformations in the inertial frame are also different, i.e. $T_{in}^{A_t^{(1)},A_{t+1}^{(1)}} \neq T_{in}^{A_t^{(2)},A_{t+1}^{(2)}}$.

the state of a body at time step $t$ in absence of other bodies. A triple of transformations $T^{B_t,O}$, $T^{B_{t-1},B_t}$ and $T^{B_t,B_{t+1}}$ provide a complete description of a state of a body in some fixed frame of reference $O$ which accounts for a constant or stationary environment. Similarly, transformations $T^{A_t,O}$, $T^{A_{t-1},A_t}$ and $T^{A_t,A_{t+1}}$ provide such a description for some other body with frame $A$.

The state of a system consisting of three bodies with frames $A$ and $B$ in some constant environment with frame $O$ can be described by the six transformations as it is shown in Figure 1, where $T^{A_t,O}$ has been replaced by a relative transformation $T^{A_t,B_t}$.

The prediction problem can be stated as: given we know or observe the starting states and the motion of the pusher, $T^{A_t,A_{t+1}}$, predict the resulting motion of the object, $T^{B_t,B_{t+1}}$. This is a problem of finding a function:

$$F : T^{A_t,B_t}, T^{B_t,O}, T^{A_{t-1},A_t}, T^{B_{t-1},B_t}, T^{A_t,A_{t+1}} \longrightarrow T^{B_t,B_{t+1}} \quad (1)$$

Function $F$ is capable of describing all possible effects of interactions between rigid bodies $A$ and $B$, providing their physical properties and applied net forces are constant in time, in the limit of infinitesimally small time steps. Furthermore, it can be approximately learned from observations for some small fixed time interval $\Delta t$ between time steps.

In this work, we will focus on robotic manipulations that are performed relatively slowly, hence we assume quasi-static conditions, and ignore all frames at time $t-1$. This conveniently reduces the dimensionality of the problem, giving a simplified function, $F_{qs}$:

$$F_{qs} : T^{A_t,B_t}, T^{B_t,O}, T^{A_t,A_{t+1}} \longrightarrow T^{B_t,B_{t+1}} \quad (2)$$

The behaviours of interacting bodies represented by rigid body transformations as in Figure 1 are independent of their poses with respect to some inertial frame $I$ [9]. Therefore instead of using inertial frame-dependent transformation $T_{in}^{A_t,A_{t+1}}$, one can represent object transformations in the object body frame (see Figure 2). The body frame transformation $T_{body}^{A_t,A_{t+1}}$ is obtained by moving instantaneous frame $A$, so that at time $t$ it overlaps with inertial frame $I$. Given some instantaneous object frame $A_t$ at time $t$, and the transformation $T_{in}^{A_t,A_{t+1}}$, one can obtain transformation $T_{body}^{A_t,A_{t+1}}$ in the body frame (via a similarity transform):

$$T_{body}^{A_t,A_{t+1}} = (T^{I,A_t})^{-1} T_{in}^{A_t,A_{t+1}} T^{I,A_t} \quad (3)$$

where we have made use of the identities $T^{I,A_{t+1}} = T_{in}^{A_t,A_{t+1}} T^{I,A_t} = T^{I,A_t} T_{body}^{A_t,A_{t+1}}$.
Given a transformation in the body frame, instantaneous object frame $A_t$ at $t$ and using Equation (3), transformation $T_{in}^{A_t,A_{t+1}}$ in the inertial frame is given by:

$$T_{in}^{A_t,A_{t+1}} = T^{I,A_t} T_{body}^{A_t,A_{t+1}} (T^{I,A_t})^{-1} \quad (4)$$

In further discussion we will retain subscripts $in$, but suppress subscripts $body$, and assume that all transformations $T^{X,Y}$ are transformations in the body frame $X$ obtained using a similarity transform $T^{X,Y} \equiv T_{body}^{X,Y} = (T^{I,X})^{-1} T_{in}^{X,Y} T^{I,X}$.

Since the prediction problem is posed as finding a function, we can now apply our function approximator of choice. In this paper we use LWPR [10] - a powerful method applied widely in robotics.

## III. Learning Global and Local Experts as Density Estimation

Having now formulated prediction as a function approximation problem, in this section we recast it as a density estimation problem. The motivation for this is that prediction learning using functions $F$ or $F_{qs}$ is limited with respect to changes in shape and type of manipulation.

Consider a 2D projection at time $t$ of a robotic finger with global frame $A_t$, an object with global frame $B_t$, and the constant global frame $O$ (Figure 3). We can identify local frames $A_t^l$ and $B_t^l$, rigidly attached to small local planar surface patches at the contact point, or the points of closest proximity on the object and finger. We define the global information to be the information about changes of the pose of the whole object, whereas the local information is specified by changes in the local frames $A_t^l$ and $B_t^l$.

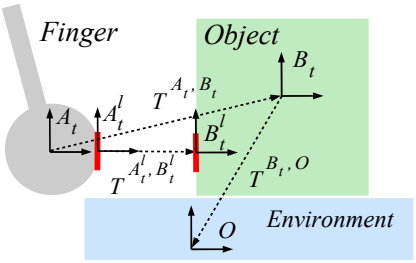



Fig. 3. 2D projection at time $t$ of a robotic finger with global frame $A_t$, an object with global frame $B_t$, and a ground plane with constant global frame $O$. Local frames $A_t^l$ and $B_t^l$ describe the local shape of the finger and an object at their point of closest proximity.

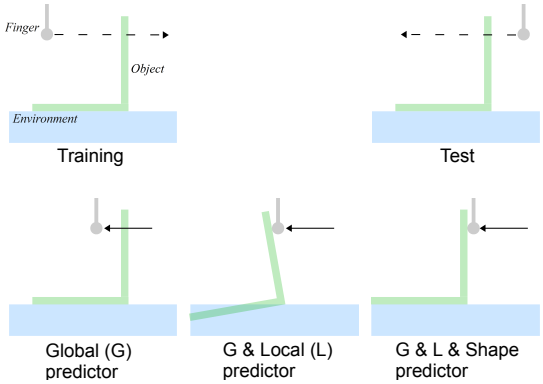



Fig. 4. Schematic diagram (2D projection of 3D scene) in which an object (of L-shaped cross-section) on a supporting surface is pushed by a robotic finger. Various predictors are trained solely on forward pushes (top left), but tested on backwards pushes (top right). The top panels show the push trajectory for the training and test phases, whereas the bottom panels show the outputs from three types of predictor in the test phase. A predictor comprised of just a global expert will fail to generalize, and will predict that the object does not move as the finger passes through it (bottom left). Adding a local expert will stop the finger penetrating the object, but does not guarantee that the predicted object motion will respect other impenetrability constraints (bottom middle). Finally, using an additional 'local shape' expert attached to the base of the object, a physically plausible motion is obtained (bottom right).

In order to combine both global and local information, one can incorporate contact information represented by transformations $T^{A_t^l,A_{t+1}^l}$ and $T^{A_t^l,B_t^l}$ directly into the domain of function $F_{qs}$. This, however, would significantly increase the dimensionality of the function's domain. Instead, we recast the mapping $F_{qs}$ as a conditional probability $P_{qs}(T^{B_t,B_{t+1}}|\cdot)$, i.e. a probability density over rigid body transformations of the object [8]. This reformulation allows us to combine the global and local information as a *product of densities* to approximate $P_{qs}$, so that (schematically, for some normalisation constant $N$)

$$P_{qs} \approx N \; P_{global} \; P_{local} \tag{5}$$

where

$$P_{global} \equiv P_{global}(T^{B_t,B_{t+1}}|T^{A_t,A_{t+1}}, T^{A_t,B_t}, T^{B_t,O}) \tag{6}$$

$$P_{local} \equiv P_{local}(T^{B_t^l,B_{t+1}^l}|T^{A_t^l,A_{t+1}^l}, T^{A_t^l,B_t^l}) \tag{7}$$

denote the global and local density functions or "experts" [8]. The densities $P_{global}$ and $P_{local}$ factorise the conditioning variables of $P_{qs}$, and hence manage the complexity of incorporating more information into the predictor.

The above global and local densities encode information about which candidate rigid body transformations are more or less feasible for each frame of reference respectively. However, once we form the product of these two densities, only transformations which are feasible in both frames will have high probability in the resulting combined distribution.

The rationale for introducing global and local experts, instead of using a straightforward function approximation, can be explained by considering a backward-push experiment as shown in Figure 4.

The configuration of finger and object during a backward push is very different to those present in a training set consisting only of forward pushes. A predictor comprised of just a global expert will fail to generalize to a new push direction that differs markedly from any observed in the training set for the expert. However, by also using the local expert $P_{local}$, the predictor can learn that the finger does not penetrate the object after contact. Any candidate motion preferred by the global expert will be 'vetoed' by the local expert if impenetrability is violated. Nevertheless, there are other constraints on the object motion, such as the ground plane, which are not encoded by the local expert. To model these other facts about possible object motion requires the use of additional experts as described in the next section.

Returning to the formal development, we now consider the relations between transformations expressed in the body frame of the local patches and corresponding transformations in the inertial frames. For coordinate frames as shown shown in Figure 3, from object rigidity and using Equation (3) we have:

$$T^{A_t^l,A_{t+1}^l} = (T^{I,A_t^l})^{-1} T_{in}^{A_t,A_{t+1}} T^{I,A_t^l} \tag{8a}$$

$$T^{B_t^l,B_{t+1}^l} = (T^{I,B_t^l})^{-1} T_{in}^{B_t,B_{t+1}} T^{I,B_t^l} \tag{8b}$$

where $I$ is the inertial frame. $T^{A_t^l,B_t^l}$ can be determined directly from the shape frame:

$$T^{A_t^l,B_t^l} = (T^{I,A_t^l})^{-1} T_{in}^{A_t^l,B_t^l} T^{I,A_t^l} \tag{9}$$

For the finger-object scenario, a prediction problem can then be defined as finding that transformation $\widetilde{T}_{in}^{B_t,B_{t+1}}$ in the inertial frame which maximises the product of the two conditional densities (experts) (6) and (7):

$$\widetilde{T}_{in}^{B_t,B_{t+1}} = \underset{T_{in}^{B_t,B_{t+1}}}{\operatorname{argmax}} \left\{ P_{global} \; P_{local} \right\} \tag{10}$$

where the similarity transforms (3) (in frame $B_t$) and (8b) must be used to evaluate $P_{global}$ and $P_{local}$ for a given $T_{in}^{B_t,B_{t+1}}$.

Starting with some initial state of the finger $T^{A_0}$ and object $T^{B_0}$, and knowing the trajectory of the finger $A_1, \ldots A_T$ over $T$ time steps, one can predict a whole trajectory of the object $B_1, \ldots B_T$, by iterating the prediction obtained from

Equation (10). That is, the output of the prediction at time $t$ is used as input to the prediction for the next time step.

## IV. Incorporating information from additional experts

In addition to learning how an object moves in response to a push, it is desirable to incorporate learned information about the inherent tendencies of parts of an object to move in various directions with respect to the environment or other objects, regardless of whether the object is being pushed or not. This additional information may help when predicting the motion of a previously unseen object, or the response to a novel push direction (Figure 4), because it provides some prior knowledge about which kinds of motions are possible and which are not.

We can incorporate this additional information by attaching an arbitrary number of additional coordinate frames $B^{sn_t}$ to various parts of the object (Figure 5).

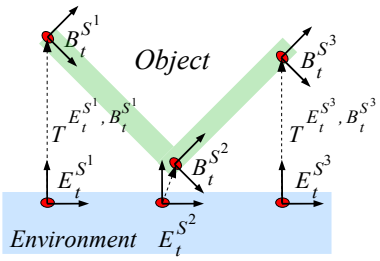



Fig. 5. Co-ordinate frames can be attached to an arbitrary number of local shapes, and local experts can be learned for each of these frames, predicting a distribution of how the frame may move next, given where it is at the present time step.

We then learn densities, also known as local shape experts, for the future motions of each of these frames. To obtain the results presented in this paper, the number and location of local shape experts on each of the different objects were determined by hand.

The local shape densities are conditioned only on their relative pose $T^{E^{S_t^k}, B^{S_t^k}}$ with respect to a corresponding pose $E^{S_t^k}$ of a patch on a ground plane at the present time step, ignoring any information about the motions of the pushing finger. For the $k$-th such frame, we estimate *the local shape conditional density*:

$$P_{shape,k} \equiv P_{shape}(T^{B^{S_t^k}, B^{S_{t+1}^k}} | T^{E^{S_t^k}, B^{S_t^k}}) \qquad (11)$$

which represents the probability density over possible rigid body transformations in the body frame of the $k$-th local contact. Analogous to Equation (10), the subsequent motion of the object in the inertial frame can be predicted as:

$$\widetilde{T}_{in}^{B_t, B_{t+1}} = \underset{T_{in}^{B_t, B_{t+1}}}{\operatorname{argmax}} \left\{ P_{global}\, P_{local} \prod_{k=1 \ldots N} P_{shape,k} \right\} \quad (12)$$

where $N$ is the number of local shape experts (Figure 6).

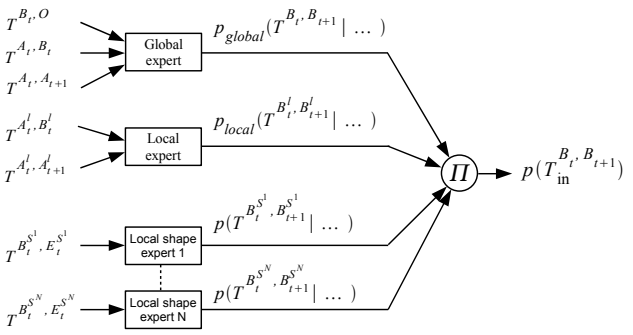



Fig. 6. Inputs and outputs of learned prediction system. The two-expert approach can be extended to include opinions from multiple local shape experts represented by coordinate frames $S^N$.

## V. Implementation

We have now presented two formulations of the prediction learning problem: 1) as function approximation, and 2) as density estimation. We have suggested that there may be an advantage to solving the density problem by applying the heuristic of a product of experts (densities).

*Regression method.* We used LWPR [10] to estimate the mapping described by Equation (2). The regression scheme was implemented using the LWPR software library [11].

*Single expert and multiple expert methods.* A variant of Kernel Density Estimation is used to approximate conditional densities in (12) (for details see [8]). The single expert method employed only a global expert (6). The density product (12) is maximised using the differential evolution optimisation algorithm [12], which has a run time proportional to the product of the population size and the number of generations used in the algorithm. The run time also scales linearly with the number of experts, the number of kernels and the number of parameters used to encode the rigid body transform.

Rigid body transformations used in both learners were parametrised by 6 numbers: Euler angles and a displacement.

## VI. Experimental study

We have tested our prediction algorithms in a number of experiments (see section C), in which a real robot arm applies pushes to various real objects. The arm has accuracy of $\pm$1mm in the region of the contacts in the reported experiments, and the predictors are trained on poses captured by a particle filter based tracker, which has pose errors of the order of $\pm$2mm frames for most frames, with up to $\pm$5mm in 5% of frames for some videos where the polyflap object is beginning to tip over. These tracking errors are significantly smaller than the average prediction errors generated by any of the predictors ($\pm$ 20 to 80mm) as well as the differences between those average prediction errors ($\pm$ 6 to 50mm).

Section D presents the results of simulation experiments, which are designed to test the ability of learned predictors to generalise in various different ways. The simulation environment usefully provides us with perfect ground-truth data

against which to evaluate predictions, and also enables a very large number of experiments with many different values of key parameters (e.g. shape of pushed objects). Replication of the experiments in Section D on the real robot is planned future work.

Section C shows that the virtual environment (using NVIDIA PhysX) does not replicate the physical properties of the real world perfectly. We hand tuned the parameters of the physics engine to best fit the world, and in principle this could also be done automatically. However, we have found that even when optimised, the parameters neither correspond to their true values, nor do they generalise well. However, regardless of how well they correspond to the real world, the simulations still provide a self-consistent experimental environment within which to compare the accuracy of predictors that have been trained within that environment.

### A. Setup

Multiple experimental trials were performed, in which a robotic arm equipped with a finger performs a random pushing movement towards an object (Figure 7). In each experiment data samples are stored over a series of such random trials. Each trial lasts exactly 10 seconds, while data samples are stored every $1/15$th of a second.

For real experiments, we use a 5-axis Katana robotic manipulator [13] equipped with a single rigid finger, and the motion of pushed objects is captured using a single camera and a visual tracking algorithm [14]. Simulation experiments are carried out using the NVIDIA PhysX physics engine [15].

Local shape experts in the multiple expert method were fixed by hand to a L-shaped object (referred to as "polyflap") as it is shown in Figure 5). In the case of a box-shaped object (Experiments 3 and 5), there were 4 local shape experts fixed to the edges of a box.

The bandwidth of all distributions used in the multiple experts method as well as parameters of the LWPR regression method were tuned once by hand and kept constant throughout all the experiments.

### B. Performance measure

In all experiments, we take the output of the tracked 6D pose of a real object to be ground-truth, and compare it against predictions which were previously forecast by the learned prediction system. The vision system does not provide perfect ground-truth, yielding typical errors of around $\pm$2mm during successful tracking, or arbitrarily large errors when the track is occasionally lost. However, comparing predictions to the outputs of the tracker still provides some useful information about discrepancies in the predictor, although clearly the performance of the predictors is limited by the accuracy of the data on which they are trained. Prediction performance is evaluated as follows.

At any particular time step, $t$, a large number, $N$, of randomly chosen points $p_n^{1,t}$, where $n = 1 \ldots N$, are rigidly attached to an object at the ground-truth pose, and the corresponding points $p_n^{2,t}$ to an object at the predicted pose. At time step $t$, an average error $E_t$ can now be defined as the

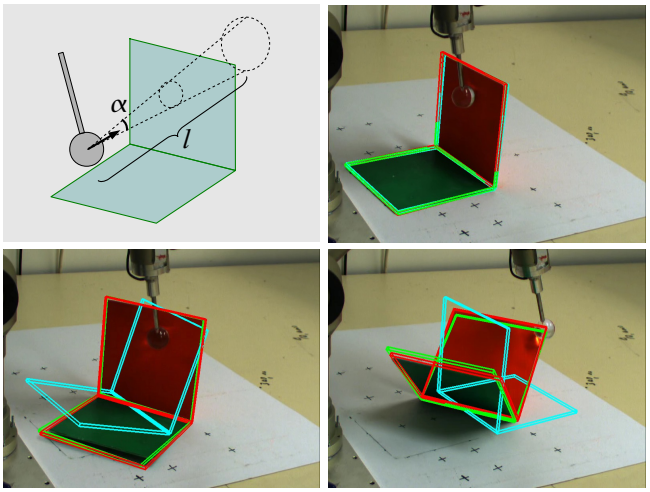



Fig. 7. A 5-DOF robotic arm equipped with a finger performs a random straight-line pushing movement of a variable length $l$=25$\pm$5 cm within a cone with angle $\alpha$=20 deg towards an object (top left). The movement begins at a random location so that every small region on the upper part of an object is equally likely to be pushed. The object behaviour can be complex and varies depending on the finger trajectory and its pose relative to the object. In the image sequence shown above, the object begins to rotate anti-clockwise (top right - bottom left) before tilting (bottom right). The red wire-frame shows the output from the vision tracking system. The green wire-frame indicates the object pose predicted by the multiple-expert learning method, while the blue wire-frame is generated by the PhysX simulator. Although the PhysX predictions are qualitatively plausible, it was virtually impossible to tune the simulator so that its predictions match reality for all training data. Note that the entire motion sequence is predicted before the physical push is initiated, without any correction from visual feedback during the push execution.

mean of displacements between points on the object at the predicted pose and points on the object at the ground-truth pose:

$$E_t = \frac{1}{N} \sum_{n=1 \ldots N} |p_n^{2,t} - p_n^{1,t}| \tag{13}$$

Note that for each robotic push action, we predict approximately 150 consecutive steps into the future, with no recursive filtering or corrector steps, hence it is expected that errors will grow with range from the initial object pose. We therefore find it more meaningful to normalise all errors with respect to an "average range", $R_t$, of the object from its starting position, defined as:

$$R_t = \frac{1}{N} \sum_{n=1 \ldots N} |p_n^{1,t} - p_n^{1,0}| \tag{14}$$

For a test data set, consisting of $K$ robotic pushes, each of which breaks down into many consecutive predictions over $T$ time steps, we can now define average error and normalised average error:

$$E_{av} = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{T} \sum_{t=1}^{T} E_t, \quad E_{av}^{norm} = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{T} \sum_{t=1}^{T} \frac{E_t}{R_t} \tag{15}$$

For each set of test data, we also report final error and normalised final error, which represent the typical discrepancy between prediction and ground truth that has accumulated by the end of each full robotic push:

$$E_f = \frac{1}{K}\sum_{k=1}^{K} |p_n^{2,T} - p_n^{1,T}|, \quad E_f^{norm} = \frac{1}{K}\sum_{k=1}^{K} \frac{|p_n^{2,T} - p_n^{1,T}|}{R_T} \tag{16}$$

Note that both normalised errors have no units.

We performed 10-fold cross-validation where at the beginning of each experiment all the trials are randomly partitioned into 10 subsets. Prediction was then subsequently performed (10 times) on each single subset, while learning (only for learned approaches) was always performed on the remaining 9 subsets of these trials. All the results were then averaged to produce a single estimation.

### C. Experiments with a real robot

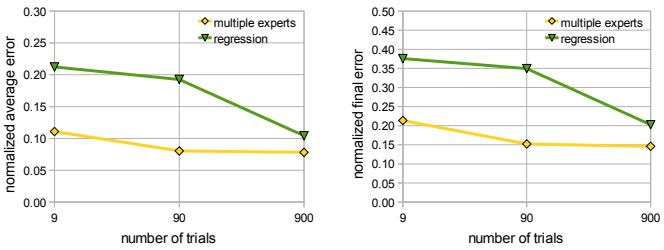



Fig. 8. Experiment 1 with a real robot and a polyflap object. Decrease in average (left) and final (right) prediction errors with increasing number of learning trials, for two different prediction methods.

*Experiment 1: comparison of learning methods for a real robot pushing a polyflap object.* We have trained the system on 9, 90 and 900 pushes of a polyflap object with a real robotic finger (Figure 7). We evaluated the performance of the multiple expert and regression methods. Figure 8 shows that the average and final prediction error decreases with increased number of trials used in learning for both tested prediction methods. The multiple expert method performed reasonably well, even when trained on as little as 9 example pushes. The method performed particularly well with 90 learning trials, as local experts successfully prevented the predictor from violating impenetrability constraints that were frequently violated by the regression method. However, the performance of the multiple expert method did not significantly improve with 900 learning trials. One of the reasons for this is that the visual tracking system is far from perfect. The tracking often contains significant errors, and the quality of tracking is not pose-independent. For example, cases of tipping and toppling movements are particularly difficult to track, so that the prediction system does not always have sufficiently accurate training data to precisely learn all possible motions.

Additionally we obtained predictions using the NVIDIA PhysX physics simulator, with parameters hand-tuned to match the real system. Figure 9 presents a comparison of the physics simulation and the learned predictors (trained on 900 trials). Clearly, the physics simulator is unable to match predictors trained in a real experiment, even though the real training data contains significant errors due to occasional failures and inaccuracies of the vision system. In particular, the physics simulator has difficulty modelling the frictional interactions of the real world, and often is unable to accurately simulate a rotational movement of the object.

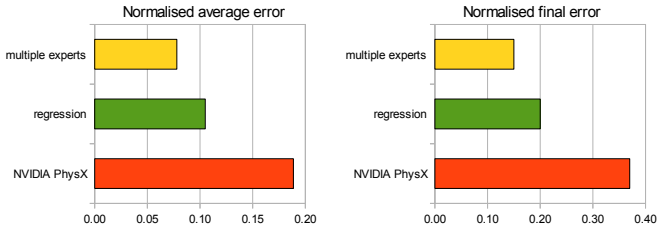



Fig. 9. Experiment 1. Physics simulation is unable to match the performance of learned predictors which have been trained in real experiments.

*Experiment 2: comparison of learning methods for a real robot pushing a small box.* We have trained the system on 9, 90 and 450 pushes of a small box object with a real robotic finger. Figures 11 and 12 show examples of the multiple expert method making accurate predictions of the box motion when it topples and when it rotates under manipulative pushes. As with Experiment 1, the learning converges within a few hundred example pushes. The multiple expert performed reasonably well, even when trained on as little as 9 example pushes.

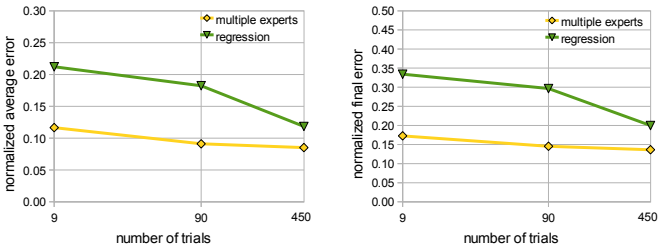



Fig. 10. Experiment 2 with a real robot pushing a small box object. Decrease in average (left) and final (right) prediction errors with increasing number of learning trials, for two different prediction methods.

### D. Experiments in a virtual environment

*Experiment 3: extrapolative generalisation of pushing directions.* In this experiment, a virtual robotic arm applied random orthogonal and oblique pushes to the outside of a polyflap which were then used in training. In contrast, the system was tasked to make predictions for previously unencountered pushes – those applied to the inside surface of the polyflap (thus pushing in the opposite direction to the training pushes). We consider this to be a test of "extrapolative" action generalisation, in that the push directions used

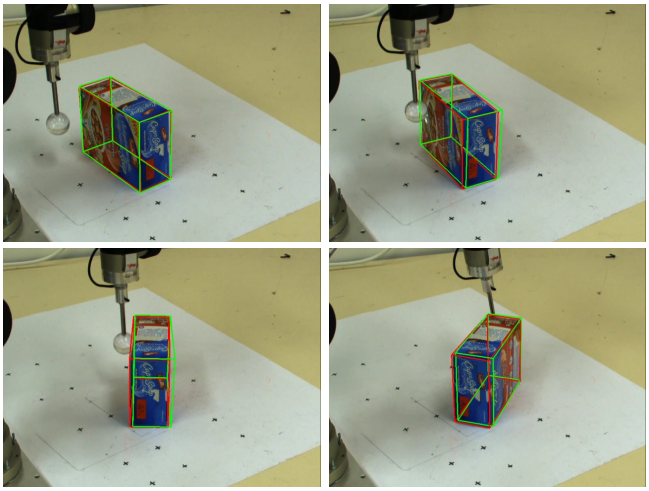


Fig. 11. Experiment 2: accurate predictions of the motion of a small box, as it translates and rotates under a manipulative push from a real robot. The green wire-frame indicates the predicted object pose; the red wire-frame shows the tracked pose from the vision system.

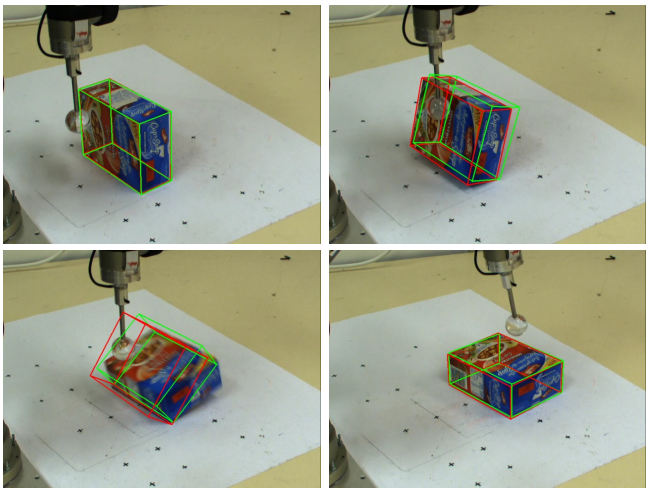


Fig. 12. Experiment 2: accurate predictions of the motion of a small box, as it topples over, under a manipulative push from a real robot. The green wire-frame indicates the predicted object pose; the red wire-frame shows the tracked pose from the vision system.

in testing are all qualitatively different from those used in training – the test push directions do not lie in the same region of data covered by the training examples. The regression and single expert methods failed to predict the polyflap behaviour, and gave physically implausible predictions in which the fingertip penetrated the polyflap (Figure 13). In contrast, the multiple expert method gave a relatively accurate prediction, in which even inaccurate portions of the object trajectory were still physically plausible, and did not violate basic physical constraints on object behaviour such as impenetrability (Figure 13). Note that the motion model is entirely learned – there was no pre-programming of Newtonian laws of motion, gravity, the ground plane, or impenetrability constraints.

*Experiment 4: extrapolative generalisation to novel shapes.* In this experiment, the predictors were trained on a polyflap, but were then tasked with predicting the motion of a box - a new shape which had never been encountered in training. This is a test of "extrapolative" shape generalisation. The multiple expert method correctly predicts the direction of motion of the box, and makes a physically plausible prediction (but fails to predict that the box should topple over) (Figure 14). In contrast, the regression and single expert methods constantly violate physics, predicting that the fingertip will penetrate right through the box.

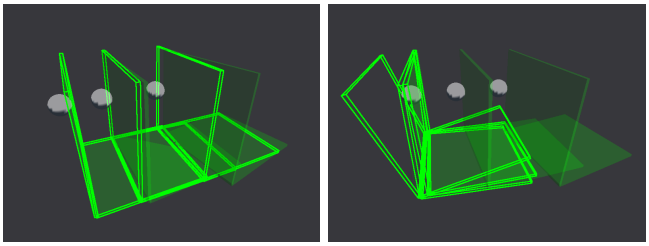


Fig. 13. Experiment 3. Simulation experiment in which predictors are trained only on pushes applied to the inside of the polyflap (moving from right to left in the figure), but are then tested on pushes applied to the outside of the polyflap (i.e. from left to right). The multiple expert method (left panel) predicts a rightwards movement, that comes close to the true motion, does not violate impenetrability, and is physically plausible. In contrast, the regression method (right panel) erroneously predicts that the fingertip (shown as a ball) will pass right through the polyflap. The ground-truth and the predicted poses are shown as solid and wire-frame shapes respectively.

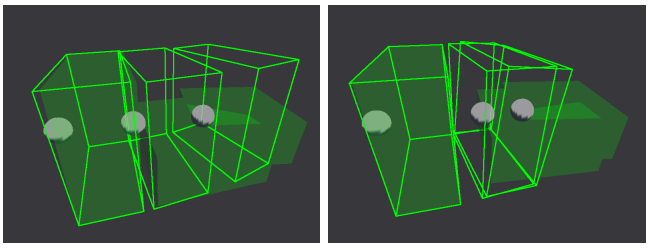


Fig. 14. Experiment 4. Simulation experiment in which predictors have been trained on a polyflap, but tasked with making predictions for a box. The multiple expert method (left panel) predicts a motion which is erroneous (i.e. fails to predict toppling in this case), but is in the correct direction, is physically plausible, and does not violate impenetrability constraints. In contrast, the regression method (right panel) violates impenetrability constraints, as does the single expert method (not shown). The ground-truth and the predicted poses are shown as solid and wire-frame shapes respectively.

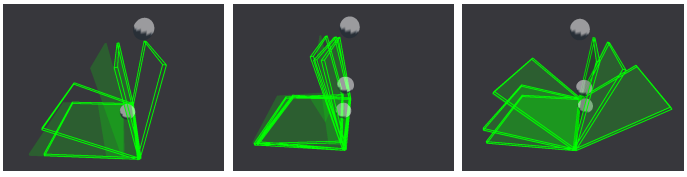


Fig. 15. Experiment 5 reveals limitations of the regression and single expert methods, which fail to predict the motion of a polyflap when subjected to a downward push (left panel). The multiple expert method can cope well with this kind of shape variation (middle and right panels). The ground-truth and the predicted poses are shown as solid and wire-frame shapes respectively.

*Experiment 5: interpolative generalisation to novel shapes.* This is a virtual experiment, in which all training and

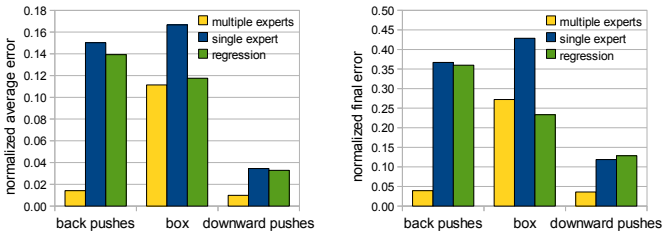



Fig. 16. Experiments 3, 4 and 5. Action generalisation errors for back pushes (Experiment 3), shape generalisation errors for a box (Experiment 4), and downward pushes (Experiment 5).

| Exp | Trials | Predictor | $E_{av}$ [m] | $E_{av}^{norm}$ | $E_f$ [m] | $E_f{}^{norm}$ |
|---|---|---|---|---|---|---|
| 1 | 900 | Multi exp. | **0.021** | **0.078** | **0.036** | **0.146** |
| | 900 | Regression | 0.027 | 0.104 | 0.050 | 0.206 |
| | n/a | PhysX | 0.044 | 0.189 | 0.083 | 0.372 |
| 2 | 450 | Multi exp. | **0.023** | **0.085** | **0.037** | **0.136** |
| | 450 | Regression | 0.032 | 0.118 | 0.056 | 0.200 |
| 3 | 900 | Multi exp. | **0.005** | **0.014** | **0.015** | **0.039** |
| | 900 | Single exp. | 0.054 | 0.150 | 0.143 | 0.367 |
| | 900 | Regression | 0.051 | 0.139 | 0.141 | 0.360 |
| 4 | 900 | Multi exp. | **0.042** | **0.111** | 0.103 | 0.272 |
| | 900 | Single exp. | 0.064 | 0.167 | 0.169 | 0.429 |
| | 900 | Regression | 0.045 | 0.118 | **0.093** | **0.233** |
| 5 | 900 | Multi exp. | **0.002** | **0.009** | **0.008** | **0.036** |
| | 900 | Single exp. | 0.007 | 0.035 | 0.023 | 0.119 |
| | 900 | Regression | 0.007 | 0.033 | 0.026 | 0.129 |

TABLE I

COMPARATIVE PERFORMANCE OF TESTED PREDICTORS. $E_{av}$ AND $E_f$ ARE MEASURED IN METRES. THE OTHER MEASURES ARE UNITLESS AS EXPLAINED ABOVE.

testing data involve polyflaps constructed from two square flanges. Random shape variation consists in varying the angle at which the two square flanges are connected along a common edge. This shape variation is very significant - dramatically changing the finger-object contact relations. For example, depending on small changes in the angle of the flanges, the same push from above might cause the entire object to move either leftwards or rightwards (Figure 15). The experiment reveals limitations of the regression and single expert methods. Since these methods do not encode information about the contact variability, they do not generalise well in situations where small changes in shape can cause significant and qualitative changes in the resulting motion, even when the robotic push is the same. In contrast, the product of experts technique copes much better with this kind of shape generalisation. We consider this a form of "interpolative" generalisation task, in that the test and training shapes are qualitatively similar and the range of test shapes can be considered to be spanned by the range of training examples. The results are presented in Figure 16.

## VII. CONCLUSIONS

This paper has presented several methods by which a robot can learn to predict the motions of a rigid object that will result from manipulative pushing actions. We have shown how regression can be used to efficiently learn the overall "global" motion of a body. We have further shown how multi-modal distributions of local parts of the motion can be learned by Kernel Density Estimation, and how many of these "local" experts can be combined as a product of densities, significantly extending the capabilities of the system with respect to generalization.

This is the first work of which we are aware, in which explicit predictions of 3D object motions under push manipulation are enabled without hard coding of Newtonian physics and physical constraints, but rather by learning based on simple proprioceptive sensing and visual observations of manipulated bodies. The learning approach significantly outperforms approaches based on physics simulators which often model real world interactions poorly, and which rely on physical parameters which may not be known. Furthermore, the proposed multiple expert approach provides a degree of generalisation with respect to changes in shape and applied actions.

## VIII. ACKNOWLEDGMENTS

We gratefully acknowledge the support of EU-FP7-IST grants Nos 248273 (GeRT) and 215181 (CogX).

## REFERENCES

[1] M. T. Mason, *Manipulator grasping and pushing operations*. PhD thesis, MIT, 1982.
[2] K. Lynch, "The mechanics of fine manipulation by pushing," in *IEEE Int. Conf. on Robotics and Automation*, pp. 2269–2276, 1992.
[3] M. A. Peshkin and A. C. Sanderson, "The motion of a pushed, sliding workpiece," *IEEE Journal on Robotics and Automation*, vol. 4, pp. 569–598, 1988.
[4] D. J. Cappelleri, J. Fink, B. Mukundakrishnan, V. Kumar, and J. C. Trinkle, "Designing open-loop plans for planar micro-manipulation," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 637–642, 2006.
[5] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini, "Learning about objects through action-initial steps towards artificial cognition," in *IEEE International Conference on Robotics and Automation, 2003. Proceedings. ICRA'03*, vol. 3, 2003.
[6] B. Ridge, D. Skocaj, and A. Leonardis, "Towards learning basic object affordances from object properties," in *Proceedings of the 2008 International Conference on Cognitive Systems*, 2008.
[7] L. Paletta, G. Fritz, F. Kintzler, J. Irran, and G. Dorffner, "Learning to perceive affordances in a framework of developmental embodied cognition," in *IEEE 6th International Conference on Development and Learning, 2007. ICDL 2007*, pp. 110–115, 2007.
[8] M. Kopicki, J. Wyatt, and R. Stolkin, "Prediction learning in robotic pushing manipulation," in *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pp. 1–6, 2009.
[9] M. Kopicki, *Prediction learning in robotic manipulation*. PhD thesis, University of Birmingham, 2010.
[10] S. Vijayakumar, A. D'souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, 2005.
[11] S. Klanke, S. Vijayakumar, and S. Schaal, "A library for locally weighted projection regression," *The Journal of Machine Learning Research*, vol. 9, pp. 623–626, 2008.
[12] R. Storn and K. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
[13] Neuronics AG, "Katana user manual and technical description." http://www.neuronics.ch, 2004.
[14] T. Mörwald, M. Zillich, and M. Vincze, "Edge tracking of textured objects with a recursive particle filter," in *Proceedings of the Graphicon 2009*, (Moscow, Russia), 2009.
[15] NVIDIA PhysX, "Physics simulation for developers." http://developer.nvidia.com/object/physx.html, 2009.

## A.6 Videos

### A.6.1 Dora 2

`http://www.youtube.com/watch?v=0QcmSDZR-c4`

This video shows and explains a typical course of action when Dora is trying to find an object. It illustrates how the probabilistic and structural default knowledge is exploited and more efficient behaviour is generated by the switching planner.

### A.6.2 George 2

`http://cogx.eu/results/george/`

This video explains the main principles implemented in George. It describes the operation of the robot from a system perspective with the emphasis on three learning mechanisms and shows fragments of the corresponding mixed-initiative dialogue.