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

DR 7.1:
Analysis of a robot that achieves tasks under partial information

⟨cogx@cs.bham.ac.uk⟩

Due date of deliverable:	Mar 31 2010
Actual submission date:	Mar 31 2010
Lead partner:	BHAM
Revision:	final
Dissemination level:	PU

This deliverable reports on insights gathered in two of the key scenarios in CogX: Dora and George. These two complementary systems are the testbed for the implementations of our models of self-understanding and self-extension. We report on the release strategy, the current system architectures, and summarise the results of studies and evaluation runs carried out to assess these integrated systems.

1 Introduction

In the CogX project we strive to build integrated robotic systems that self-understand and self-extend and study these in a systemic way. This document is the first of a series of deliverables that will take a systemic perspective and discuss the robots' system architectures, their specific contributions to the overall project's goals and theories, and their evaluation with respect to demonstrated robot behaviour and validation of theories.

Our general roadmap in systemic evaluation is based on an interweaved release and experiment strategy. This document reports on the development towards an experimental integrated system conducted in year 1 and its subsequent analysis. This strategy for the systemic evaluation will be retained throughout the project: A major release of the system is targeted for the annuals reviews, its detailed analysis is carried out afterwards.

In the first year, work has been carried out in three scenario named *Dora*, *George*, and *Dexter*. Our integration effort in this first year were focused on the first two which already integrate quite a number of different components and functionalities developed in different WPs. Therefore, these

two scenarios are subject to discussion in this document. They have been designed to be complementary, focusing on dedicated aspects of the CogX theories and accordingly emphasising different work packages. Both systems are fully integrated using the same framework (CAST) and schema (PECAS) which has been extended to meet the requirements of CogX's systems. The scenario *Dexter* is up to now fully covered by work in WP2. Hence, its analysis is fully reported in that WP's deliverables.

The document is structured according to the two key scenarios. The structure for each scenario is similar. First, we briefly summarise the scenario and their underlying motivation. Afterwards, we outline the general evaluation strategy and report on the system architecture and abilities of the release to be studied. Finally we present and discuss some of the insights gathered in the systemic evaluation.

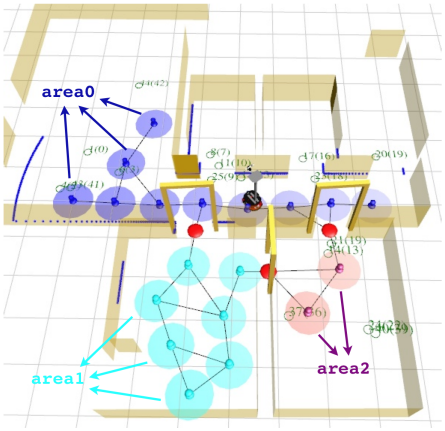
2 Dora the Explorer

Dora is our mobile robotic demonstrator that focuses on spatial representation. In year 1 we focused on partial information by explicitly representing *knowledge gaps* and taking actions to fill these gaps, resulting in a "curious exploring robot" that is driven by its general objective to learn more about the world. Consequently, all our evaluation efforts are gathered around studies of the exploratory behaviour of Dora. Fig. 1 shows the mobile robot platform and also sketches parts of the spatial representation that robot is building up incrementally.

2.1 Scenario

The general theme of the demonstrator "Dora" is self-understanding and -extension with respect to representations of space. Dora is a mobile robot with an understanding of spatial structures, categories, and functions. Dora's use case is motivated by the vision of robots in people's homes that fulfil certain tasks, such as fetch-and-carry. Therefore a robot requires a very rich and most complete knowledge of its environment. Teaching a robot this knowledge implicitly is however a very tedious task for a human. Dora instead, given an incomplete tour of an indoor environment, is driven by motivations to probe the gaps in her spatial knowledge and extend it. Dora implements curiosity-driven self-extension with regard to its hierarchical spatial knowledge as detailed in the attached paper [3]. While Dexter is focused on manipulation, the Dora scenario is designed as a test case for the various aspects and challenge of mobile robot that has to cope with incomplete knowledge. While in year 1 interaction with humans was minor in Dora, human interaction partners are considered as potential sources of knowledge in the coming years of CogX. This will enable concepts of the two demonstrators Dora and George to converge in the longer run. The

(a) The Dora platform: a P3 mobile robot base with a custom-built super-structure and different sensors.



(b) Visualisation of Dora's map of a partially explored environment. Coloured disks denote *place* nodes (colour indicates segmentation into different rooms, `area0,1,2`). Small green circles represent opportunities for spatial exploration (*placeholders*). Red nodes indicate places where doorways are located.

Figure 1: Dora the Explorer: a robot system for goal-driven spatial exploration.

scenario in year 1 focused on generating behaviour to self-extend by spatial exploration. The robot basically is able to *explore* places it has not been to before to more completely map the environment and to *categorise* rooms by exploiting ontological knowledge about objects it searches for in these rooms.

2.2 Evaluation Approach and Framework

In the integration of Dora we follow a release strategy of one stable release of the system each year corresponding to the review dates. In between such two stable releases we are committed to at least one release that is related to the regular spring or summer school organised by CogX. The releases are each also available as a Live CD distribution to document and preserve project progress. Besides these major releases we follow a continuous integration strategy as many of the scientific question we are investigating can only be studied in an integrated manner.

2.2.1 Description of Release Yr 1

It is key to the approach in CogX that all research is instantiated in the scenarios and consequently in year 1 already many conceptual contributions have made it into the integrated system of Dora as dedicated *subarchitecture* that roughly correspond to WPs:

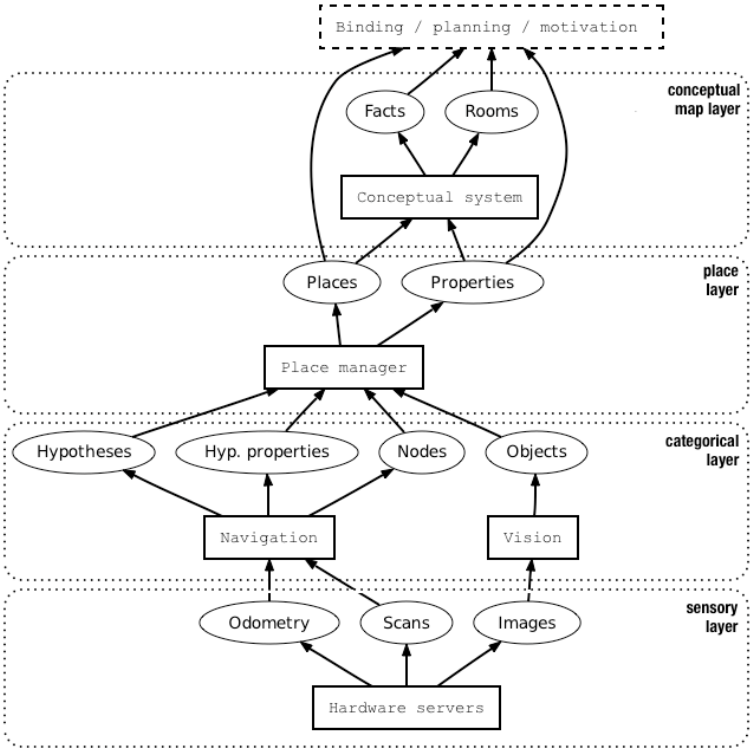


Figure 2: Sketch of the spatial representation as implemented in Dora

subarchitecture	**WP**
spatial.sa & coma.sa	3
planner.sa	1&4
binder.sa	6

Work of other work packages and subarchitectures is as well relevant for Dora but was not in the focus of the scientific progress to be demonstrated in this integration scenario.

The year 1 release which constitutes the basis for the analyses presented in this paper comprises the following abilities developed in dedicated WPs:

- A first instantiation of the four layer spatial representation (WP3)[6]. The *sensory layer* provides continuous low-level readings from sensors. Readings are clustered and classified quantitatively in the *categorical layer*. The results are used in the *place layer* to form discrete Places and Placeholders, and their associated properties. The components of the *conceptual map layer* (coma.sa) perform qualitative reasoning over these abstractions. Firstly, the conceptual map layer segments interconnected Places into rooms and maintains room instance representations. Non-monotonic reasoning is employed here to encounter errors in perception and to assure a consistent representation. Sec-

ond, the reasoner tries to infer more special categories for rooms, e.g., office or kitchen. One novelty is that the association between room categories and salient objects is established through the "locations" table of the OpenMind Indoor Common Sense database. For further details about the spatial representation please refer to the attached document [9].

- *Binding* serves to fuse information from different modalities, into singular amodal representations [5]. In the current implementation of Dora it provides a unified view of the system's state to support reasoning and planning. The approach of probabilistic binding itself is key to the George scenario, however the framework is also used in Dora to provide the same technological framework for both scenarios to ease future integration.

- Continual planning and monitoring: The planner developed as part of the efforts in WP4 is a multi-agent *continual planner*[1], capable of replanning and execution monitoring. It is essential that a continual approach is used when planning in interactive robot systems such as Dora. Such an approach is required to handle changes in state, changes in goals, and the sensing and execution failures which naturally occur in such systems.

- A framework for goal generation and management: As part of the *planning.sa* developed in WP1 of CogX we introduced an architectural concept of goal generation and management referred to as *motivation framework*. With our focus on self-extension, it is this layer that generates epistemic goals by analysing the spatial representation, initiates planning algorithms in order to achieve epistemic goals, and finally monitors the execution of plans. In the context of the current exploration system of year 1 it decides on a behavioral level which exploration goals to pursue next. Basically, we consider two types of goals: exploration to extend the spatial coverage of the map, or exploration to increase the amount of categorical instance information in the conceptual map. Details about goal generation and management can be found in the attached document [2].

With the current implementation of Dora being focused on spatial representations two types of knowledge gaps give rise to epistemic goals: yet unexplored places and detected rooms that are not yet categorised. Accordingly, the robot can have to simple epistemic goals: explore a place or categorise a room.

Dora's system architecture is composed of five of the subarchitectures running all on one Laptop computer on the autonomous robot, cf. Fig. 1(a). The composition is sketched in Fig. 3. The diagram is adopted from UML

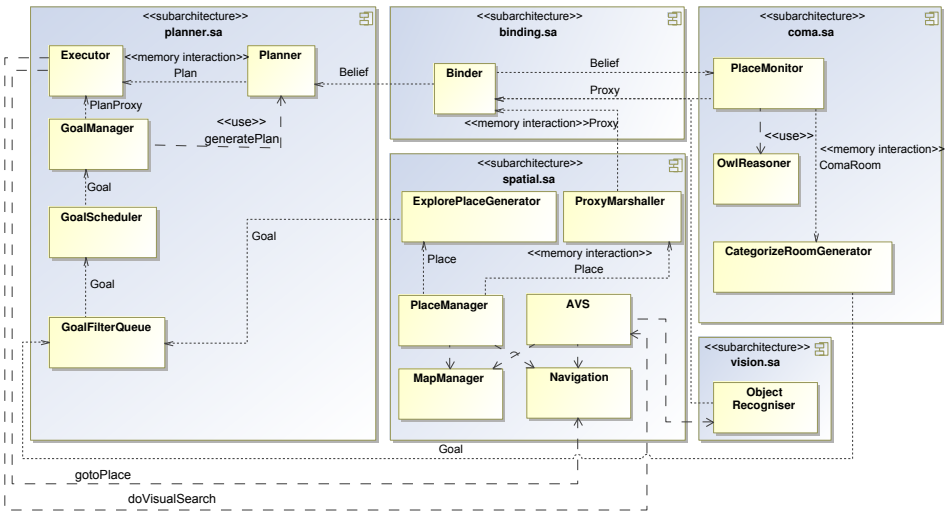


Figure 3: Dora system architecture. For clarity, all memory interactions are not depicted as information flow mediated through working memories in the subarchitectures but as directed dotted connections of sources and sinks. The dashed lines represent synchronous request-reply calls to components by mutually modifying memory structures.

2.0 specification and illustrates the information flow between components, and across subarchitectures. Most of the flow is realised by interactions with the working memories in an event-driven manner as proposed by CAS (CoSy Architectural Schema). The diagram does however not include all components. We focus here on those that are required to understand the architecture facilitating self-understanding and -extension, disregarding those that can be seen as support services and sensor abstraction only.

Taking a closer look at the actual processes in the current implementation of Dora, we can see how the interactions between components work. Fig. 4 pictures the activity that the robot goes through from the detection of a knowledge gap to its filling. The example illustrated in the figure is corresponding to "explore place" only, but the structure is quite similar for "categorise room". It starts with *spatial.sa* hypothesising a new place and thus generating a *Place* in the working memory that is marked as being hypothetical. This generation triggers *binding* and *ExplorePlaceGenerator* to create a *Belief* about this place and an epistemic *Goal* to explore this place. After the motivation-related components have filtered and scheduled the generated goal, the planner is triggered to generate a plan to actually achieve it. The *Executor* then executes the actions of the plan. One action will be to navigate towards the placeholder which will – in this example – make it explored. This update is again propagated through the working memory, resulting in the goal to be removed and the belief being updated

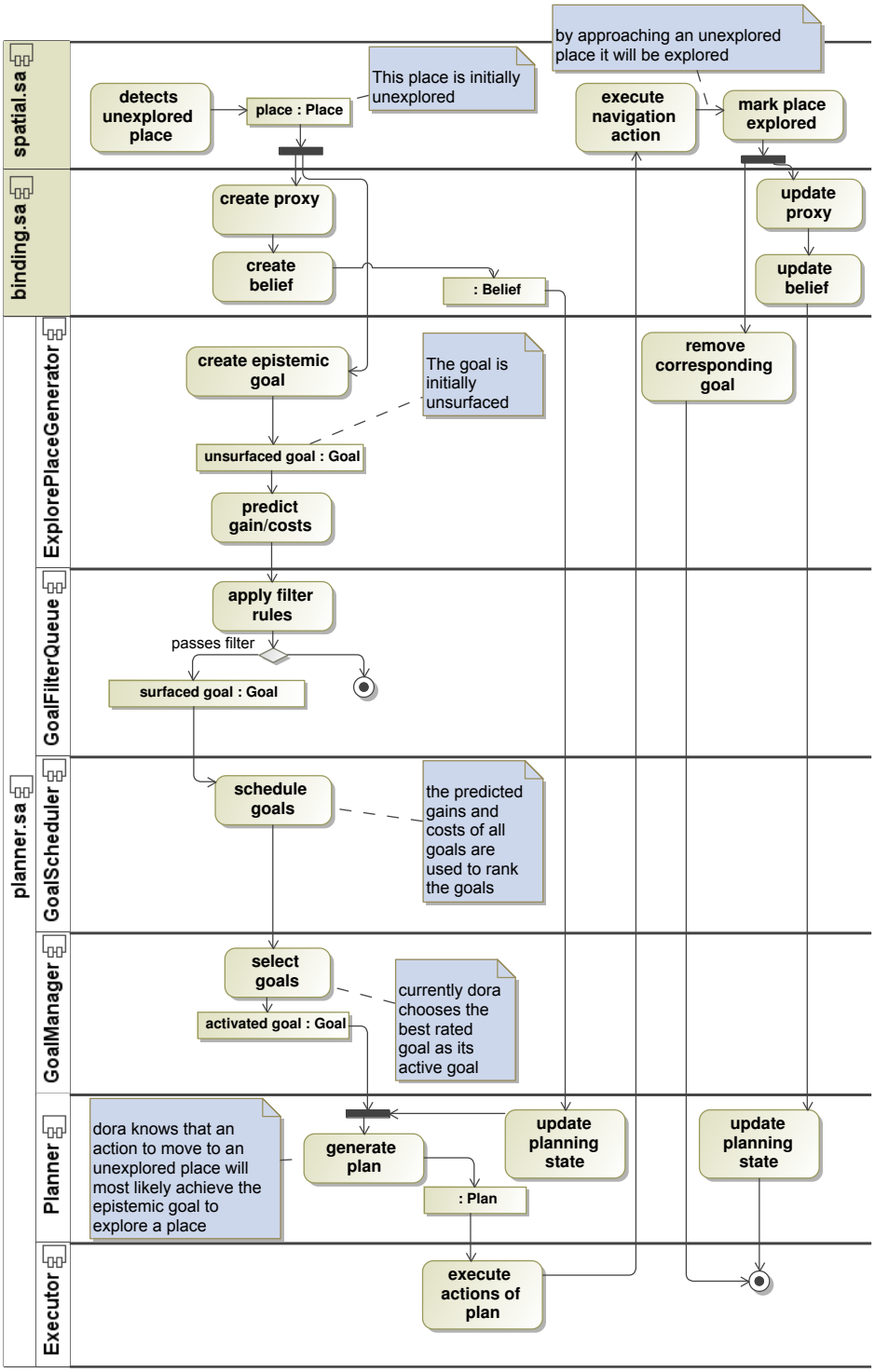


Figure 4: Activity diagram illustrating the "path" of a knowledge gap from its generation to its filling.

asynchronously.

2.2.2 Evaluation Schemes

We generally employ a staged evaluation scheme going hand in hand with the release management outline before.

component tests (CoT): For most of our research-relevant components we also developed "dummy components" that can substitute the original one. This allows us to individually test relevant components. Using CAST's working memory based integration [4] the substitution of a dummy component by the real one is done transparently. Additionally, a generic memory recorder and player have been developed to transparently simulate information flow and test on a component level. For some components, standardised test (e.g. using JUnit) have been implemented, too.

systemic evaluation in controlled environments (SEICE): In order to facilitate meaningful and structured development and evaluation, the Dora system is concurrently integrated in the simulation environment provided by the *stage* toolkit and the real robots at all institution involved in research on Dora. The simulation environment which emulates the sensors and motors of our robot allows to study systemic aspects of our systems in more efficient way but still being very close to reality. Our integration approach allows us to run the same composition of components – the same system architecture – both in reality and simulation with only very minor changes on a driver and sensor level. SEICE allows us to derive meaningful, also *quantitative*, measures of overall system performance. Batch experiments in this scheme are reproducable and more comparable than in reality while still being a close-to-reality system.

systemic evaluation in reality (SEVIRE): The ultimate test for any integrated robot system is of course its application in real world. However, the effort for real robot runs is quite high considering safety aspects, recharge and setup times, and the fact that experiments cannot be parallised. Accordingly, our strategy is to use SEVIRE to (i) prove the general (qualitative) applicability of our approach in real worlds, i.e. by video recordings, and (ii) to explore and discuss the limitations of insights gathered in SEICE when it comes to real world problems. Additionally, aspects of real interaction (with humans) and the real non-determinstic nature of real world systems can only be studied in a SEVIRE setting.

2.2.3 Tool Support

When studying specific aspects in a systemic context or even striving to deal with the complex assessment of the overall system performance adequate tool support is key to success. Powerful introspection, logging, and analysis functionalities to allow for comprehensive *post-mortem* analysis (after a number of experiments have been carried out) needed to be implemented. In CogX we employ a combination of CAST [4][1]-innate logging, standard tools (log4j and log4cxx), and XML-based selection and transformation techniques (based on XQuery) in conjunction with MatLab or similar tools to compute quantified measures of system behaviour and performance for dedicated purposes as illustrated also in [2].

2.3 Insights Gathered from Release 1

In this report we focus on insights that have been gathered by studying the integrated robot systems (SEICE and SEVIRE evaluations). Component evaluations are subject to WP-specific deliverables. The experiments we conducted in the first year were tailored to the analysis of the exploration behaviour of Dora and studied the course of actions the robot undertook to achieve self-extension accounting for non-monotonicity, incompleteness, and uncertainty of the representations being build up.

In our aim to study the exploratory behaviour of Dora in a structured way also quantitatively two systemic studies in controlled settings have been conducted, both being reported in publications [2, 9]. They comply to the SEICE approach outlined before. Accordingly, we created a simulated environment of a real place (part of DFKI building, upper floor, Saarbrücken, Germany) to test the systems in (sketched in Fig. 5). We decided for SEICE because we wanted to have quantitative measures in a more controlled fashion and repeated experiment under the same conditions several times. With the SEICE approach we still can study the effects of uncertainty and non-determinism as we only simulated low-level sensors and actuators which can be modelled to have realistic operation characteristics (e.g. a typical detection rate of only 90% for the simulated object visibility). Hence, even in a simulated environment the system's behaviour is therefore not inherently deterministic. In the two studies we were interested in the following questions:

1. How does the robot extend its spatial representations in a consistent way by its generated behaviour?

2. Is the system robust enough to explore a realistically sized environment? Robustness is not only considered in terms of software or hard-

[1] CAST is the CoSy Architecture Schema Toolkit that is employed as integration framework for all the systems in CogX.

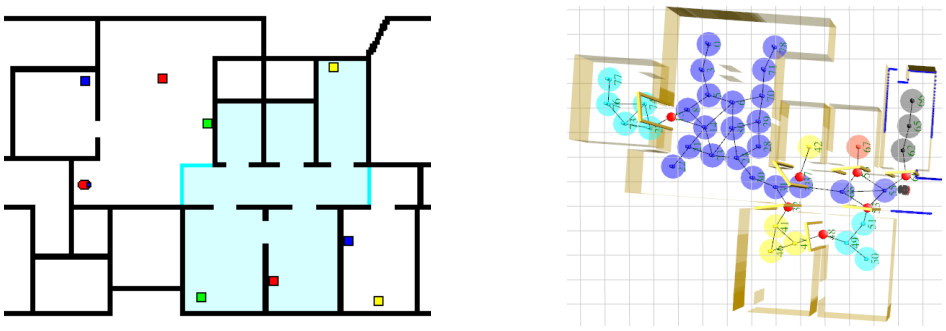


Figure 5: Stage simulation model used in the experiments (l) and screenshots of the visualisation tool acquired during one of the three runs of the exploration study (r). In the second study a smaller section of the whole environment is used (highlighted in cyan colour). The small coloured boxes depict objects in the environment used for room categorisation.

ware failures (though this is relevant as well), but also on a task level, i.e., is the robot able to generate appropriate behaviour.

3. Is our model of non-monotonic inference and self-extension adequate to the exploration task we study?

4. What is the effect of the goal management mechanisms and how does it impact the self-extending behaviour of the robot?

5. What is the planning effort in the integrated system when filling knowledge gaps?

6. What are the bottlenecks and drawbacks identified in the system so far?

In the following, we briefly summarise the two studies we carried out to analyse how Dora achieves tasks under partial information and how she extends her knowledge.

2.3.1 Exploration and non-monotonic reasoning

One consequence of the uncertainty and partiality of the observations Dora is dealing with is that the map building process is non-monotonic. Structural and conceptual abstractions may need to be reconsidered in the light of new evidence acquired during the active exploration. In this study we were interested in the behaviour of the robot tailored to autonomously explore yet unknown spaces. So in this experiment the robot only attained to *explore* goals and did not consider to categorise any of the rooms it detects during this exploration. We were basically interested in the overall

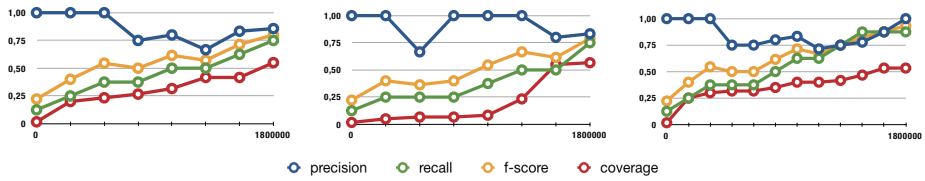


Figure 6: Plots for precision, recall, balanced f-score and coverage of each of the three experimental runs. The Y-axis shows the normalised values for precision, recall, balanced f-score, and coverage (0–1). The X-axis is time, in milliseconds.

coverage the robot achieves in exploration (thus, evaluating the spatial representation and the goal generation from gaps in spatial knowledge) and the non-monotonicity in the formation of rooms derived from the analysis of connected places. This analysis is tailored to question 1-3 mentioned above. Details of this analysis can be found in the attached document [9].

To evaluate the coverage an exploration of the full environment depicted in Fig. 5 yields, we determined a gold standard of 60 Place nodes to be generated in order to fully and densely cover the simulated environment. We achieved this by manually steering the robot to yield a complete coverage, staying close to walls and move in narrow, parallel lanes. We performed three runs with the robot in different starting positions, each time with an empty map. Each run was cut-off after 30 minutes. The robot was then manually controlled to take the shortest route back to its starting position. Fig. 6 illustrates that in all three runs the robot at the end of the autonomous exploration explored more than half of the number of places we defined as gold standard (denoted as relative *coverage* in the diagrams). Though this appears to be a rather low ratio it is still sufficient to cover most of the space and comprises all the rooms as exemplary shown in Fig. 5 for one of the runs (on the right side). It shall be noted that it is not the goal of the robot to explore the map similarly to the gold standard but to acquire a representation that does not have any more knowledge gaps detected in the spatial model. Furthermore, the 30 minutes cut-off of the experiment left some hypotheses still being unexplored. The employed goal management scheme developed in WP1 schedules according to a trade-off of information gain and associated costs of a goal (in this case of exploring a place hypothesis). This strategy led to an exploration behaviour that was not tailored to the exploration of all hypotheses in the 30 minute time. However, the analysis of individual runs and the global assessment of the coverage indicate a suitability of our integration of the spatial model and the goal selection with regard to the generation of hypotheses about place and the subsequent generation of epistemic goals from these hypotheses to drive the robot's behaviour.

Also meaningful is the analysis of the *precision*, *recall*, and *f-score* with

respect to the detection and consistent representation of rooms in Dora as shown in Fig. 6. It can be seen that the accuracy (balanced f-score) of the representation is monotonically increasing towards a high end result (0.8, 0.79 and 0.93, resp.). The increases and decreases in precision during the individual runs are due to the introduction and retraction of false room instances generated by, i.e. false detection of gateways. Recall can be interpreted as coverage in terms of room instances. After 30 minutes the exploration algorithm yielded a relatively high recall value (0.75, 0.75 and 0.875, resp.), i.e., most of the rooms had been visited. For details and further discussion of the results please cf. [9].

From this analysis we can conclude that the incremental self-extending behaviour of the spatial representation (including places and rooms) in Dora is a first successful implementation of an integrated approach for an exploration task. The system is functionally and technologically robust enough to explore realistic environments.

2.3.2 Selection of and planning for epistemic goals

In order to study the goal generation and management processes and the continual planning more in detail a second study was conducted, again following the SEICE approach exploring a smaller section of the environment shown in Fig. 5. This time, the robot's task has been extended. Not only did the robot have goals to explore hypotheses about places it could visit but it also tries to *categorise* the rooms it detected using the ontological reasoning about the objects found in this rooms. The focus of this study was therefore designed to approach basically questions 4 and 5 outlined above, hence to investigate the interplay of planning and goal management. Extending the first study which only considered one class of epistemic goals, namely to explore as yet unexplored places, this study investigated how the goal selection scheme chooses between the different classes of goals (namely *explore place* and *categorise room*) generated from the gaps in the spatial representation in the course of the experiment. Fig. 7 pictures the course of action in one of the 15 runs conducted in this setting. It illustrates when a new knowledge gap of a dedicated class has been detected and when a gap has been filled after the goal has been scheduled and the corresponding actions have been executed. This diagram helps us understanding the goal selection strategy. It is apparent that the robot interleaves different goals reflecting their respective information gain and costs.

To investigate the interplay of planning and goal management we implemented two configurations of the system. The first configuration explicitly encodes its two drives (to explore space and categorise rooms) as the single planning goal:

```
(and (forall (?p - place) (= (explored ?p) true))
     (forall (?r - room)  (kval 'robot' (areaclass ?r))))
```

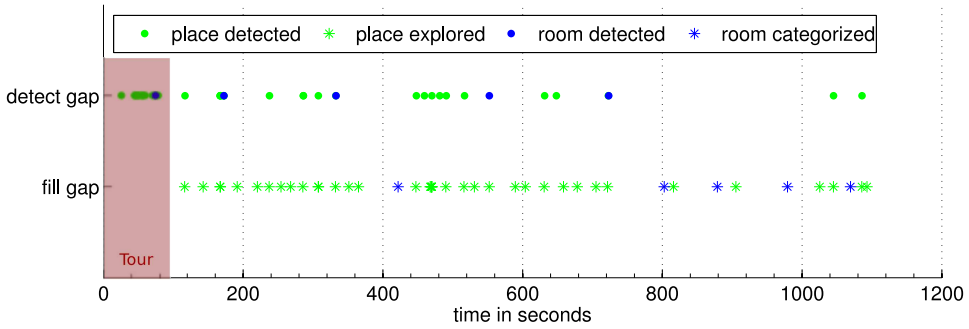


Figure 7: Exemplary course of action for filling knowledge gaps in a real run.

This goal, literally interpreted as "explore all places and know a category for every room", is passed to the continual planner directly as one complex goal that needs to be achieved. In this configuration, termed *conjunct goal set (CGS)*, all system behaviour is determined by the continual planner in response to this unchanging goal. The planner continuously monitors the system's state, triggering replanning if any relevant state changes occur (e.g. unexplored places appearing on the spatial WM). In this set we introduced the complete exploration of a room (all places being in a room) as a precondition for the categorisation of that particular room. The second configuration, termed *managed goal set (MGS)*, employs our implementation of the goal generation and management framework of WP1. In this configuration the planner is fed the individual goals selected by the management mechanisms. For details on the management mechanisms please cf. [2].

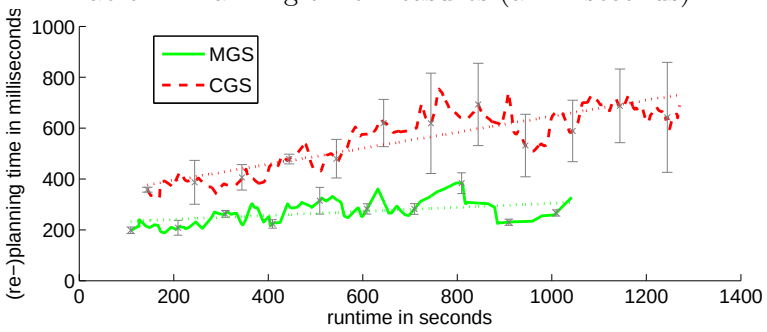
In the experiments, we were able to show that in both configurations the robot was able to do the exploration task. It proves that the approach of continual planning with its monitoring of the system state accounts well for the open-endness of the exploration task. However, in CGS plans have to be revised with new gaps appearing and being removed. In average, in the CGS configuration the robot had to generate new plans 79.0 times during the runs. With 31.1[2] the number is significantly lower in the MGS configuration where the individual goals are much more simple. Not only does the complexity of the conjunct goal affect the number of replanning actions but it also has an impact on the overall time spend planning as indicated by table 1 which summarises the time measurements of planning in the two configurations. The differences between the averaged timings taken for the two configurations are statistically significant with $p < 0.0001$ in Mann-Whitney testing for all measures shown in the table.

Fig. 8(a) underpins the hypothesis that planning becomes harder with

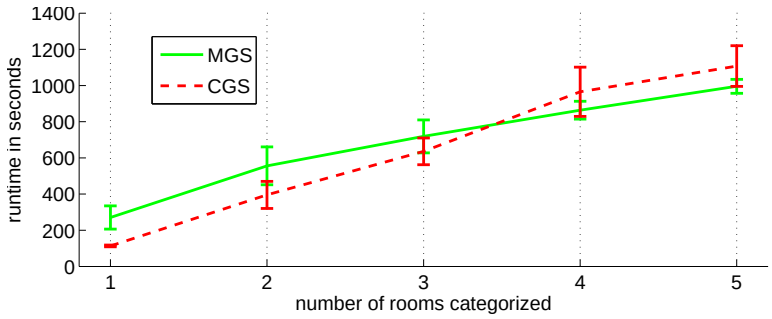
[2]These numbers include not only replanning but also the first invocation of planning to generate a plan for a given goal. In CGS there is only one initial planning invoked and all later ones are replanning.

	CGS	**MGS**
avg. time per planning call	0.621 s	0.292 s
avg. time spent on planning	48.843 s	8.858 s
avg. time spent on actions	1000.923 s	757.312 s

Table 1: Planning time measures (all in seconds).



(a) Averaged planning time during a system run.



(b) Running time when rooms have been categorized.

Figure 8: Dora timing information.

the exploration progressing. The planning time is averaged at discrete time steps across all the successful runs of each setup. The error bars indicate the standard error in averaging. It can be seen that the progression over runtime is different in the two cases. While the trend, indicated by a linear fitting shown as a dotted line in Fig. 8(a), is a shallowly included line for MGS, a steeper increase in average planning time can be seen for CGS. This steeper increase can be associated with the increasing size of the planning problems the CGS configuration faces as Dora's knowledge increases: planning for all possible goals over a larger and larger state space becomes increasingly difficult.

Concluding, we can say that the rather limited complexity of the current exploration task in the year 1 release of Dora can well be mastered in both configurations. However, with goal management we have a framework that ensures tractability and advanced goal selection in our scenarios. It is also interesting to take a look at the differences in the resulting behaviours of

the robot. While the CGS configuration only considers the costs of the actions to achieve the overall goal the goal management scheme uses the predicted heuristic information gains and the costs to schedule the single goals. This results in different behaviour as indicated in Fig. 8(b). The plot depicts the time (y-axis) by which a given number of rooms have been categorised (x-axis), averaged over all successful runs. It shows that the MGS configuration takes longest to categorise the first two rooms before speeding up to be the fastest to complete four and five rooms. In our runs, the MGS configuration did not choose to immediately categorise the room it start its autonomous operation in. Instead it chose to visit unexplored places that were attributed with a high information gain, even though the precondition for categorising this first room was fulfilled. In contrast, the CGS configuration always categorises the first room immediately as soon as all its places have previously been explored because it deems this to be the cheapest approach to achieve the overall goal. This illustrates the potential for future investigation of self-extending behaviour. An informed strategy for goal selection taking into account a currently executed plan while considering alternative and opportunistically interleaving task-oriented behaviour with other epistemic goals contributing to potential future tasks is one ambition in the scenario of Dora.

2.3.3 Further aspects and conclusions

The Dora release of year 1 has successfully demonstrated all relevant pieces of software working in an integrated system. The level of integration achieved allowed for both quantitative and qualitative evaluation of a fully integrated system. The quantitative studies were supported by a simulation environment (SEICE) which allows us to achieve better repeatability and more (statistically) significant results in shorter times. However, the system runs robustly in real world enabling us to demonstrate it to other researcher to gain further qualitative insights from their comments and criticisms. In summary, researchers were quite impressed by the level of integration achieved with the system. Quite many liked to see more and different tasks being achieved by the robot autonomously. They asked for more informed model of information gain and commented on the limited rationale of the robot's behaviour. Generally, the approach we have chosen has been well received and we have been encouraged to continue in this direction.

Besides numerous such demonstrations at CogX-involved institutes Dora participated as a finalist in the BCS Machine Intelligence Competition[3] in December 2009. The year 1 release of Dora is also accepted as a demo for the AAMAS 2010 conference [3]. We are further on committed to continuous integration of the Dora system at all research sites that are in-

[3] `http://www.comp.leeds.ac.uk/chrisn/micomp/2009entries.html`

volved. So, systemic testing in the SEICE and SEVIRE settings will continue and intensify. A video of a SEVIRE run in real world is available at `http://cogx.eu/results/dora`.

The studies conducted with Dora have provided us with a deeper understanding on the challenges ahead. The insights gained have already been compiled into the updated implementation plan for the next release reported in the supplementary deliverable D.7.X1: "Integration Scenarios - Implementation & Evaluation Plan Year 2".

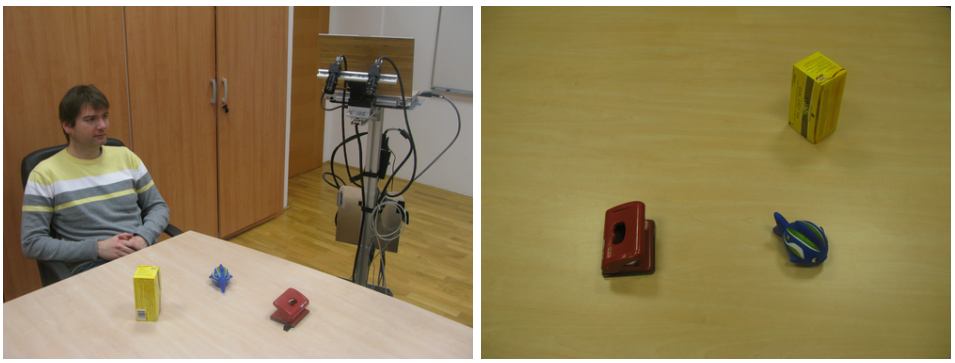
The framework is now in place and is well understood in terms of the studies presented here. It allows us to study different strategies in task planning and execution with goal management, to further investigate our spatial representations, and in general proceed towards more informed self-extension and autonomous task execution under partial information.

3 Curious George

The George scenario has been designed to demonstrate, monitor, and show progress on the development of the integrated system for *learning the association between visual features of an object and its linguistically expressed properties*. The main goal is, therefore, to integrate the developed vision routines, learning and recognition competencies, dialogue capabilities, as well as different kinds of representations and belief models in an overall system.

3.1 Scenario

The robot operates in a table-top scenario, which involves a robot and a human tutor (see Fig. 9(a)). 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. In the beginning the robot does not have any representation of object properties, therefore it fails to recognize the objects and has to learn. To begin with, the tutor guides the learning process and teaches the robot about the objects. After a while, the robot takes the initiative and tries to detect its own ignorance and to learn autonomously, or asks the tutor for assistance when necessary. The tutor can supervise the learning process and correct the robot when necessary; the robot is able to correct erroneously learned representations. The robot establishes transparency and verbalizes its knowledge and knowledge gaps. In a dialogue with the tutor, the robot keeps extending and improving the knowledge. The tutor can also ask questions about the scene, and the robot is able to answer (and keeps giving better and better answers). At the end, the representations are rich enough for the robot to accomplish the task, that is, to correctly describe the initial scene.



(a) Scenario setup. (b) Observed scene.

Figure 9: Continuous interactive learning of visual properties.

3.2 Evaluation Approach and Framework

3.2.1 Description of Release Yr 1

According to the Technical annex, the main goal in year 1 was set to *develop a learning mechanism for learning basic visual concepts grounded to signals. The system should be able to build associations between features extracted from input visual data (colour and depth images) and visual attributes (e.g., colour, shape) and to connect them using language in a dialogue with the tutor. Adequate mechanisms for unlearning should be investigated as well.* The main goal was, therefore, to integrate approaches developed in WP 5 into an overall system, exploiting the methods developed in other workpackages (mainly WP 2, WP 6, as well as WP 7 and WP 1).

The George Y1 system consists of three subarchitectures that have been mainly developed within the following WPs:

subarchitecture	**WP**
visual.sa	2&5
binder.sa	6
comsys.sa	6

Fig. 10 depicts the relationships between these subarchitectures and also shows some of the components that are involved.

The main competencies of the robot that have been developed within the WPs and integrated in the robot George are the following:

- *Vision routines.* We have developed the low-level vision routines that provide visual information, which is used by higher levels for learning, recognition and verbalisation. We have implemented a bottom-up attention mechanism based on detection of the parts of the scene that are sticking out from the main surface. These spaces of interest are then filtered and further processed and segmented providing object candidates.

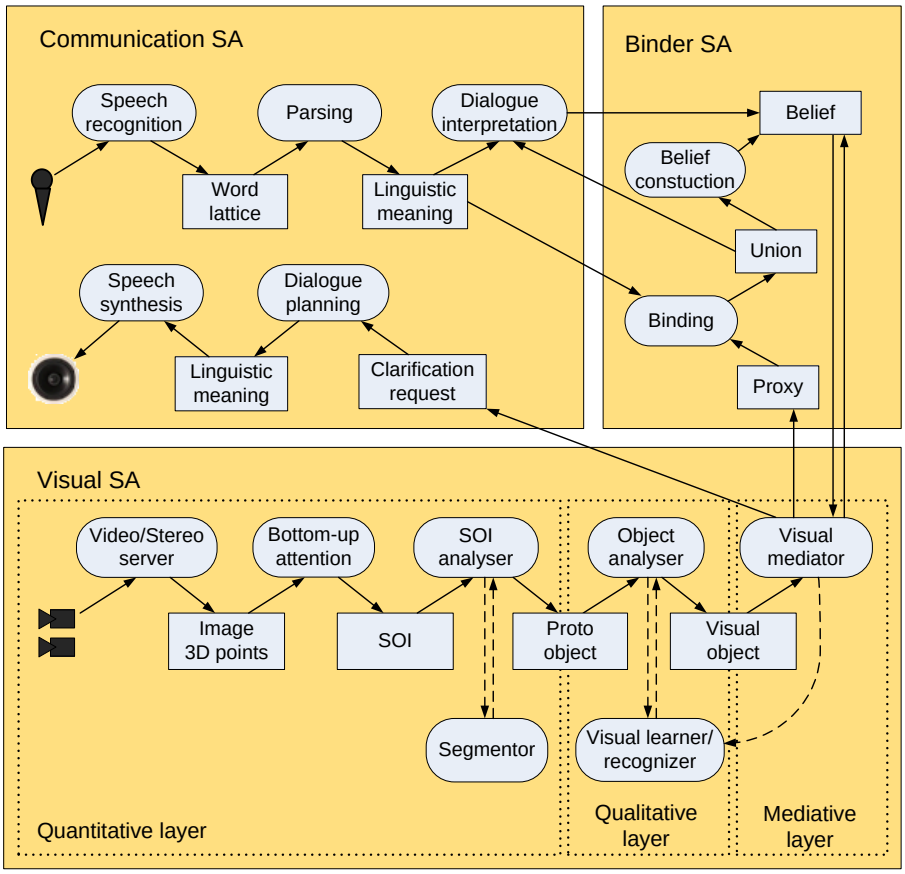


Figure 10: Architecture of the George system.

- *Online learning and unlearning.* We have developed novel methods for online learning and unlearning that are based on Kernel Density Estimation and serve as a main representation for learning and recognition of visual properties. During online operation, a KDA-based multivariate generative model is continually maintained for each of the visual concepts and for mutually exclusive sets of concepts the feature subspace is continually being determined allowing the construction of a Bayesian classifier, which is then used for recognition of object properties.

- *Binding.* A new binder has also been developed. The task of the binder is to decide which information originating from different modalities belong to the same real-world entity, and should therefore be merged into a belief. These beliefs integrate the information included in the perceptual inputs in a compact representation. They can therefore be used directly by the deliberative processes for planning, reasoning and

learning.

- *Situated dialogue.* Situated dialogue provides one means for a robot to gain more information about the environment. A robot can discuss what it sees and understands with a human. Or ask about what it is unclear about or would like to know more about. We have integrated these abilities, since they are an important part of interactive learning. The communication subsystem operates with the beliefs in the binder and fulfills the clarification requests that are received yielding a mixed initiative dialogue for learning visual properties.

A detailed description of the system is available in the attached paper [7]. Here, we just very briefly describe the processing flow, i.e., what happens when a new object is introduced to the scene[4]? First, from a stereo pair of images, a colour image and a 3D point cloud are obtained. Then, using the bottom-up attention mechanism, spaces of interest (SOIs) are detected [10]. SOIs are analysed and segmented and proto-objects are created. They are processed further by an Object analyser, which recognizes the object's visual properties. A Visual Mediator then packs the visual information and creates a *vision proxy* in the Binder. The Binder binds the visual information with information from other modalities and a belief is created from the obtained multi-modal information. The beliefs can also be altered by the communication subsystem through dialogue processing. In the current implementation of the robot, a keyboard is used as the input device, however speech recognition could be used instead. The Visual Mediator also monitors the beliefs in the binder, waiting for learning opportunities. When such an opportunity is spotted, a learning instruction is sent to the learner which updates the models. When the Visual Mediator is uncertain about recognition results, it can send a clarification request. The communication subsystem forms a corresponding question, and the human's answer is then used to update the models.

3.2.2 Evaluation Schemes

From the description of the system and the processing pipeline it is evident that the subsystems are closely connected to each other and that for a reliable performance of the system it is crucial that all the individual components perform reliably and that the integration is done correctly. We evaluated the developed system in two different ways.

Component tests (CoT): We tested individual competencies of the robot independently by simulating other competencies. Individual subarchitectures and even individual components within a subarchitecture were tested in this manner.

[4]The robot can be seen in action in the video, which is accessible at http://cogx.eu/results/george.

There are two main reasons for such testing schema. First, due to the high complexity of the system, it is necessary to test individual components independently before they are integrated in the system, so as not to introduce additional sources of non-reliability. These tests are made by simulating other components using "dummy components". These components are assumed to perform correctly, so all the problems can be attributed to the component being developed and therefore they can be more easily detected and removed. Debugging in a complex distributed nonhomogeneous and asynchronous system is very painful, therefore such testing techniques using a smaller number of components are absolutely necessary. Of course, the testing protocols and the requirements for the individual components have to be made while keeping the entire system and the requirements of other components in mind.

The second reason for performing component tests is the feasibility of large scale evaluation. If we want to comprehensively analyse some of the competencies, such as interactive learning, interactive work with the system is very time consuming and impractical. Learning is a gradual process and the system has to be exposed to a number (tens, hundreds, or even more) of training examples to build reliable models. Furthermore, we would like to test different learning strategies in the same conditions and directly compare the results, which is close to impossible if the real system is used for evaluation. Therefore, we instead performed quantitative evaluation in simulation. The simulation environment uses stored images, which were previously captured and automatically segmented. We used a number of everyday objects, similar to those presented in Fig. 9. Each image, containing a detected and segmented object, was then manually labeled. In the learning process the tutor is replaced by an omniscient oracle, which has the ground truth data available. In this way the extensive tests could be automatically performed and a reliable evaluation of the proposed methods were obtained. The results of this evaluation are reported in [8] for the vision subsystem.

Systematic evaluation in reality (SEVIRE): Of course, the ultimate test for any integrated robot system is its application in the real world. Again, there are two main reasons for doing system-wide evaluation.

The first one is obvious; we have to test how well the system performs as a whole. It may turn out that the individual components perform very well in isolation, assuming an ideal input from other components. However when these assumptions are only slightly violated (and other components provide less reliable data), the performance of the particular component may degrade considerably, resulting in a

non-satisfactory performance of the system as a whole. The main goal of such an evaluation was to find the weak points of the system and to design the solution for the next release of the system. We have encountered problems, particulary in the low-level vision routines that did not perform as robustly as we had hoped. When the vision delivered data that was too unreliable, this problem propagated throughout the entire system. The analysis of this problem helped us to design a solution that would improve the visual subsystem and the system as a whole and is presented in the next subsection.

The second reason is that there are some competencies that can only be tested in the integrated system; the main functionality of these competencies arise from the integration of different components. Such competencies cannot be tested in isolation. This is also the main advantage of having such an integrated system; it enables us to perform research and to develop functionalities that could not be done otherwise. The simulated environment we mentioned above can simulate only certain situations and a simple dialogue. Therefore, more complex situations, involving complex tasks, mixed initiative dialogue, combination of task-driven and curiosity driven learning etc., are too complex and cannot be reliably simulated. For evaluation of such competencies we need to run and test the entire system. In the next subsection we will discuss, which parts of the developed system have to be improved in order to implement the aforementioned functionalities.

3.3 Insights Gathered from Release 1

In year 1, we have made several contributions at the level of individual components (modelling beliefs, dialogue processing, incremental learning), as well as at the system level (by integrating the individual components in a coherent multimodal distributed asynchronous system). Such an integrated robotic implementation now enables us to conduct system-wide research with all its benefits (information provided by other components) as well as problems and challenges (that do not occur in simulated or isolated environments). We are, therefore, now able to directly investigate the relations between individual components and analyse the performance of the robot at the sub-system and system level. This allows us to set new requirements for individual components and to adapt the components, which will result in a more advanced and robust system.

The main goal in year 1 was to set up a framework that would allow the system to process, to fuse, and to use the information from different modalities in a consistent and scalable manner on different levels of abstraction involving different kinds of representations. This framework has been implemented in the robot George, which is still limited in several respects; it operates in a constrained environment, the set of visual concepts that are

being learned is relatively small, and the mixed initiative dialogue is not yet matured. We analysed these problems and set up a plan for overcoming them.

There is a need to robustify the vision capabilities of the system and to extend the visual subsystem to work more reliably in more complex environments. We will achieve this by restructuring and extending several components in the Visual SA. This will also allow us to deal with a richer set of object properties, which will in turn enrich the dialogues and the learning process.

The current system operates in a probabilistic framework; the probabilities are attached to recognized object properties, they are used in the probabilistic binder, etc. However, the entire probabilistic framework relies on the correctness of the detection (and to some extent also the segmentation) of the individual objects in the scene. Once a miss-detection or unwanted re-detection occurs, this error is propagated throughout the system. There is a need to deal with this problem and to provide mechanisms that will prevent such error propagation and that will handle the uncertainty in object detection as well.

One additional limitation of the Year 1 system was that it did not include the Motivation SA. The Visual SA was monitoring the beliefs in the binder and waiting for learning opportunities and was also sending clarification requests directly to the ComSys SA. This is not the best solution nor from the conceptual nor from the practical point of view. The Motivation SA should play its role here, taking care of learning requests and clarification requests and mediating between vision and communication in a principled way. The motives for learning should be adequately formed and processed and the corresponding plans should be created and executed. This will also allow us to engage George in more complex dialogues and perform more complex tasks, and also to include other functionalities that are being developed in other workpackages.

Also, the current implementation of detection of incompleteness in the robot's knowledge should be improved. Together with the integration of the motivational mechanism, this should result in a more realistic and user friendly mixed initiative dialogue, which should in turn result in more efficient learning.

And finally, during the development of the individual components and their integration in the overall system, as well as during system evaluation and testing, we faced a lot of problems when debugging the errors. We have been therefore developing a toll CastControl, that will facilitate the debugging process and will make the development and testing of the integrated distributed system easier.

We believe that the presented system forms a firm basis for further development. Building on this system, our final goal is to produce an autonomous robot that will be able to efficiently learn and adapt to an everchanging world

by capturing and processing cross-modal information in the interaction with the environment and other cognitive agents. Considering insights we have garnered from the George Y1 system, we will take another step toward this final goal in year2.

4 Lessons learnt & future challenges

Looking at the integration and evaluation in retrospection we can identify some lessons learnt and identified some future challenges from a system's perspective:

- The (revised) CAST [4] framework employed in the integration proved its suitability very well. The interaction between the components of subarchitectures facilitates very well the processes required to generate behaviour to tackle the incompleteness of the representations. Any piece of new information or explicit knowledge gaps submitted to working memories is being picked up by those components that can make use of it. With the tri-lingual support of CAST (java, C++, python) the systems have also successfully integrated legacy and contributed code to achieve progress beyond the state of the art already in the first year of CogX.

- It is in the nature of event-driven systems that synchronisation of system states can be an issue. By employing the *binder* as a central mediator of consistent representation we have widely compassed these issues. Binding not only becomes a process of consolidation and aggregation, but also from an architectural point serves as a synchronisation point.

- Many components (*binder*, *speech synthesis*, *object recognition*,...) and core features (event-driven processing, binder usage,...) of the architectures are shared between both scenarios, easing future convergence of the scenarios.

- As discussed we see the stronger exchange of components between the two scenarios not as an end in itself. We expect, that the goal management scheme introduced as *motivation* will contribute to the George scenario as well and will solve some of the issue there. It conjunction with *planning* these two can decide which feature to ask for or which object to choose first for interactive learning, for instance. Similarly, Dora will benefit from the dialogue abilities and more advanced belief models adopted from George in the future.

References

[1] Michael Brenner and Bernhard Nebel. Continual planning and acting in dynamic multiagent environments. *Autonomous Agents and Multi-Agent Systems*, 19:297–331, December 2009.

[2] Marc Hanheide, Nick Hawes, Jeremy Wyatt, Moritz Göbelbecker, Michael Brenner, Kristoffer Sjöö, Alper Aydemir, Patric Jensfelt, Hendrik Zender, and Geert-Jan Kruijff. A framework for goal generation and management. In *Proceedings of the AAAI Workshop on Goal-Directed Autonomy*, 2010. **attached paper, Annex A.1**.

[3] Nick Hawes, Marc Hanheide, Kristoffer Sj, Alper Aydemir, Patric Jensfelt, Moritz Gbelbecker, Michael Brenner, Hendrik Zender, Pierre Lison, Ivana Kruijff-Korbayova, Geert-Jan Kruijff, and Micheal Zillich. Dora the explorer: A motivated robot. In *Demo Proceedings of Int. Conf. on Autonomous Agent and Multiagent Systems*, 2010. accepted for publication. **attached paper, Annex A.3**.

[4] Nick Hawes and Jeremy Wyatt. Engineering intelligent information-processing systems with cast. *Advanced Engineering Informatics*, 24:27–39, 2010.

[5] H. Jacobsson, N.A. Hawes, G.-J. Kruijff, and J. Wyatt. Crossmodal content binding in information-processing architectures. In *Proceedings of the 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, Amsterdam, The Netherlands, March 2008.

[6] Andrzej Pronobis, Kristoffer Sjöö, Alper Aydemir, Adrian N. Bishop, and Patric Jensfelt. Representing spatial knowledge in mobile cognitive systems. Technical Report TRITA-CSC-CV 2010:1 CVAP 316, Kungliga Tekniska Högskolan, CVAP/CAS, March 2010.

[7] Danijel Skočaj, Miroslav Janiček, Matej Kristan, Geert-Jan M. Kruijff, Aleš Leonardis, Pierre Lison, Alen Vrečko, and Michael Zillich. A basic cognitive system for interactive continuous learning of visual concepts. In *ICRA 2010 workshop ICAIR - Interactive Communication for Autonomous Intelligent Robots*, 2010. submitted. **attached paper, Annex A.4**.

[8] Danijel Skočaj, Matej Kristan, and Aleš Leonardis. Formalization of different learning strategies in a continuous learning framework. In *EPIROB'09*, pages 153–160, 2009.

[9] Hendrik Zender, Geert-Jan M. Kruijff, Kristoffer Sjöö, Alper Aydemir, Patric Jensfelt, Marc Hanheide, and Nick Hawes. Autonomous semantic-driven indoor exploration. In *Proceedings of the Conference*

on Intelligent Robotic Systems (IROS), 2010. submitted. **attached paper, Annex A.2**.

[10] Kai Zhou, Michael Zillich, Markus Vincze, Alen Vrečko, and Danijel Skočaj. Plane Pop-Out as 3D Attention Mechanism in a Robot Vision Domain. In *International Conference on Pattern Recognition*, 2010. submitted. **attached paper, Annex A.5**.

A Annexes

A.1 Goal Generation and Management for a Mobile Robot

Bibliography Kristoffer Sjöö, Alper Aydemir, Moritz Göbelbecker, Michael Brenner, Marc Hanheide, Nick Hawes, Patric Jensfelt, Jeremy Wyatt, Hendrik Zender, and Geert-Jan M. Kruijff: Goal Generation and Management for a Mobile Robot, submitted to *IROS 2010*

Abstract Goal-directed behaviour is often viewed as an essential characteristic of an intelligent system, but mechanisms to generate and manage goals are often overlooked. This paper addresses this by presenting a framework for autonomous goal generation and selection. The framework has been implemented as part of an intelligent mobile robot capable of exploring unknown space and determining the category of rooms autonomously. We demonstrate the efficacy of our approach by comparing the performance of two versions of our integrated system: one with the framework, the other without. This investigation leads us conclude about that such a framework is desirable for an integrated intelligent system because it reduces the complexity of the problems that must be solved by other behaviour-generation mechanisms, it makes goal-directed behaviour more robust in the face of a dynamic and unpredictable environments, and it provides an entry point for domain-specific knowledge in a more general system.

Relation to WP This paper details the approach on goal generation and management developed in WP1 and integrated as integral of WP7.

Goal Generation and Management for a Mobile Robot

Marc Hanheide, Nick Hawes, Jeremy Wyatt
Intelligent Robotics Lab
School of Computer Science
University of Birmingham, UK

Moritz Göbelbecker, Michael Brenner
Institut für Informatik
Albert-Ludwigs-Universität
Freiburg
Germany

Kristoffer Sjöö, Alper Aydemir, Patric Jensfelt
Centre for Autonomous Systems
Royal Institute of Technology
Stockholm, Sweden

Hendrik Zender, Geert-Jan M. Kruijff
Language Technology Lab
German Research Center for Artificial Intelligence (DFKI)
Saarbrücken, Germany

***Abstract*— Goal-directed behaviour is often viewed as an essential characteristic of an intelligent system, but mechanisms to generate and manage goals are often overlooked. This paper addresses this by presenting a framework for autonomous goal generation and selection. The framework has been implemented as part of an intelligent mobile robot capable of exploring unknown space and determining the category of rooms autonomously. We demonstrate the efficacy of our approach by comparing the performance of two versions of our integrated system: one with the framework, the other without. This investigation leads us conclude about that such a framework is desirable for an integrated intelligent system because it reduces the complexity of the problems that must be solved by other behaviour-generation mechanisms, it makes goal-directed behaviour more robust in the face of a dynamic and unpredictable environments, and it provides an entry point for domain-specific knowledge in a more general system.**

INTRODUCTION

For an integrated system to be described as intelligent it is usually important for it to display goal-directed behaviour. The field of AI is rich in approaches for determining what actions a system should perform next to achieve a particular goal. However, a comparably small amount of time and effort has been expended on investigating approaches for *generating* and *managing* the goals of an intelligent system. In this paper we argue that such mechanisms can have a positive impact on the behaviour of a goal-directed system, and we support our position using data gathered from an integrated robot system that can be run both with and without goal generation and management capabilities.

Our research is motivated by the goal of implementing intelligent robot assistants which can perform tasks for, and with, humans in every-day environments. The complexity and open-ended requirements of the scenarios such robots must take part in has caused us to design our systems so that they are capable of generating their own behaviour at run-time, rather than having it determined explicitly by a programmer at design-time. The remainder of this paper assumes a general approach in which a system explicitly plans about how to bring about a particular state before executing the required actions.

Goal-directed behaviour requires a system to change the world in order to attain a predetermined state-of-affairs. We refer to the disposition to bring about a particular state-of-affairs as a *drive*. A waiter working in a busy restaurant may have a drive to make as much money in tips as possible, which could produce drives to keep his customers happy by making sure their food arrives quickly, that they do not run out of drinks, and that they are happy with their food. This example shows that a system's collection of drives may have internal structure. For this work we shall ignore this structure but instead focus on the problem of how a system's drives ultimately come to be realised in behaviour. For behaviour generation approaches such as planning to be applied, it is not enough to have an abstract specification of the state-of-affairs to bring about (e.g. making sure customers are happy with their food). Instead, these approaches typically require a description of a concrete state-of-affairs (i.e. an instance of the more general type) which can be brought about from the current state (e.g. that the customers sat at Table 12 are happy with their food). Following planning terminology we refer to this description of a concrete state-of-affairs as a *goal*. We refer to the process of producing a goal from a drive as *goal generation*. It is important that systems are able to perform goal generation in domains where all possible goals cannot be determined in advance. Goal generation allows a system to be autonomous in such domains by providing the ability to react to state changes where existing drives should give rise to new goals (e.g. unexpected customers entering the restaurant).

Our busy waiter does not just have the problem of generating goals; he must also choose which goals to pursue from of the collection of goals he has previously generated. For example, for each table of customers he might have a goal to take a drink or food order, enquire if they're enjoying their meal, clear plates from the table or bring their bill. Some goals may ultimately contribute to the waiter's original drive more (e.g. goals relating to a table of particularly generous customers he knows from a previous visit) and some less (e.g. goals from customers which didn't tip last time). Some goals may be quicker or easier to achieve (e.g. taking a

drinks order), and some less so (e.g. bringing the food for a large birthday party). Some goals may be achievable together and some not (e.g. depending on whether they fully occupy the waiter's arms), and some may have more pressing deadlines than others. Reasoning about constraints such as these in order to determine which collection of goals should be pursued is a process we refer to as *goal management*. It is important to realise that goal management is not a monolithic process in which a set of goals and constraints are considered and a subset selected for action; changes in a system's state (including new goals and observations) may require the reconsideration of any previously selected goals. When you consider planning and execution failures in such a framework, it becomes apparent that goal management can also play a part in increasing system robustness, in addition to prioritising which goals should be pursued.

GOAL GENERATION & MANAGEMENT FRAMEWORK

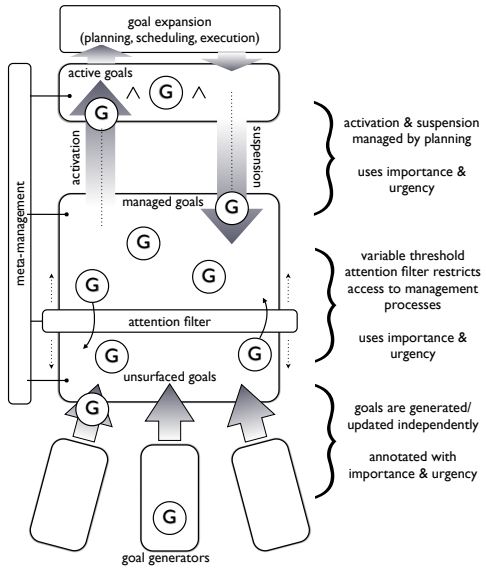


Fig. 1. The goal generation and management framework.

We would like to endow our integrated systems with the goal generation and management (GGM) capabilities of our fictional waiter. To this end we have built on the work of [1], to produce the design illustrated in Figure 1. This design is a general framework, or schema, for an architecture for goal generation and management. It specifies a collection of interacting elements which must be included in any instantiation of the framework, although the precise details of the instantiation will inevitably vary between instances. The elements of the framework are described in more detail below.

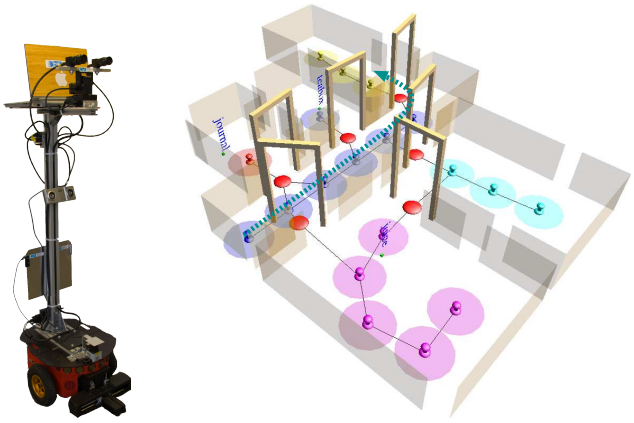
At the bottom of the framework, a system's drives are encoded as multiple *goal generators*. These are concurrently active processes which monitor the system's state (both the external world and internal representations) and produce *goals* to satisfy the system's drives. Generators can also remove previously generated goals if they are judged to no longer be appropriate. In this manner we can say that the system's drives are encoded in the goal generators (either explicitly or implicitly). We work from the assumption that as a goal passes up through the framework from a generator and influences a system's behaviour, it is inspected by processes of greater and greater computational complexity. Therefore the lower strata of the framework exist to protect these processes (and thus overall system resources) from having to consider more goals than is necessary (where this could be a contingent judgement). The main mechanism in the framework for protecting the management processes is the *attention filter*. This is a coarse barrier which uses simple, fast processing to let some some goals through to the management mechanisms whilst blocking others. Goals which make it through this filter are described as *surfaced*, thus the goals which fail to pass the filter are referred to as *unsurfaced*. A collection of management processes determine which of the surfaced goals should be combined to serve as the goals being actively pursued by the system. If a goal is selected in this way we describe it as *activated*. If a goal is removed from the set of goals being pursued by the system we refer to it as *suspended*.

In order to fulfil their roles, the filtering and management processes require information on which to base their decisions. Following the original work of [1], the framework requires that goal generators annotate each goal with a description of the goal's *importance* and *urgency*, and keep these descriptions up to date as long as the goal exists. Importance should reflect the significance of the goal to the agent (as motivated by the related drive). Urgency should reflect the necessity of achieving the goal sooner rather than later. As we shall see later, producing importance and urgency descriptions for use in such a framework is a problem in itself. In addition to these descriptions, the framework allows the management processes to use whatever approaches are required to select and maintain a set of active goals. Perhaps the minimum requirements on these processes is the ability to check whether a goal, or collection of goals, can be achieved (thus positing planning as a goal activation, as well as achievement, mechanism).

DORA THE EXPLORER

To explore the implications of this design, we have implemented the framework described above as part of a intelligent mobile robot called *Dora the Explorer*. The following sections describe the domain in which Dora operates, the subsystems which have been integrated to create Dora, and the implementation of the GGM framework in this context.

Domain: Dora is intended as a research precursor to a mobile service robot able to interact with, and perform tasks for, humans in indoor environments. Within this broad context, our particular interest lies in providing Dora with the capacity to model the limits of its own knowledge, and to plan and execute actions to extend its knowledge beyond these limits. To this end we have been investigating how Dora can represent its *knowledge gaps* (i.e. the things that it knows



(a) Dora. (b) A map constructed during a system run. The circles represent places Dora has visited.

Fig. 2. Robot and spatial environment.

it doesn't know) and how it can fill these gaps. A service robot in an office environment could have a wide variety of knowledge gaps. For example, it may know that it does not know the names, properties or functions of particular objects; the names or locations of certain people; or how to perform the actions required of it (e.g. picking up an unknown object). Although we ultimately wish to address as many of these gaps as possible, in this work we focus on two types of knowledge gap: gaps in Dora's knowledge about space, and gaps in Dora's knowledge of the function of rooms. This focus has produced an integrated robot system tailored to the problems entailed by these knowledge gaps, rather than the entire Dora domain, The design and implementation of this system is summarised in the following paragraphs.

System: The Dora robot, pictured in Figure 2(a) is based on a Pioneer 3DX with a custom super structure. The system architecture for Dora is based on the PECAS architecture [8]. PECAS divides components within a system into *subarchitectures* (SAs). SAs are collections of components plus a shared *working memory* (WM) which serves as the communication channel between components (similar to a blackboard architecture). Each WM therefore contains the representations which are shared between components in that SA. As PECAS suggests that SAs be determined by function, Dora contains SAs for the major functions required in its domain: spatial mapping; conceptual mapping; vision; and communication. These are coordinated by the SAs for planning and cross-modal fusion which are part of PECAS itself.

As our work relies heavily on planning, we will briefly describe how the planning SA in PECAS operates. Planning state is provided by the cross-modal fusion SA, which fuses representations from other SAs to produce a single, unified view of the system's knowledge. Planning goals are written to the planning WM in a format known as MAPL, the planning language used by PECAS's planner [2]. The planner itself is a multi-agent *continual planner*, capable of replanning and execution monitoring. It is essential that a continual approach is used when planning in interactive robot systems such as Dora. Such an approach is required to handle changes in state, changes in goals, and the sensing and execution failures which naturally occur in such systems. In PECAS the planner is notified of such events via changes to WM content. These changes occur asynchronously with respect to planner operation. PECAS, and any additional goal generation and management mechanisms, must be robust in the face of these events.

Dora has two SAs which contribute to its understanding of space: the spatial mapping and conceptual mapping SAs. The former of these interprets sensor data (primarily laser scans and odometry) to perform simultaneous localisation and mapping (SLAM), enabling the robot to determine its position in a local metric map. On top of these local metric maps, the spatial SA constructs a representation based on small connected regions within the the world called *places* [11]. These places represent a first order abstraction of the maps generated during SLAM and are the lowest level of spatial representation available to other SAs within the system. Furthermore, the spatial SA detects gaps in its spatial knowledge. So-called *placeholders* associated with free space represent potential places that could be explored further. The spatial SA also provides functionality for collision-free robot movement within both local metric maps and the place graph. A map produced by the spatial SA (during an experiment) can be seen in Figure 2(b). The conceptual mapping SA allows Dora to reason and abstract over entities which can appear in the maps produced by the spatial SA. It clusters collections of places which are bounded by doors into room representations. It also contains a knowledge from the OpenMind Indoor Common Sense database[1] which allows it to reason from the presence of objects in a room to the possible functional categories which could be assigned to the room (e.g. the presence of a kettle in a room may support the inference that the room is a kitchen or coffee room).

Dora is able to use this functionality to determine possible categories for rooms. Dora's vision SA contains an object recogniser which uses the Ferns algorithm [10] to identify known objects in the images it receives from Dora's cameras. The process of choosing where to capture images from in order to find objects is known as *active visual search* (AVS). Dora performs AVS by sampling the space of views of object-containing space until a coverage threshold is met. Dora is designed to assume that all non-free space on its spatial map may potentially contain objects (forcing it to look at desks, worktops and shelves in addition to featureless walls). This requires that a room must be adequately mapped prior to AVS.

Dora is implemented in C++ and Java using the CAS Toolkit [7]. It has been run on at least eight different robots (all derived from Pioneer 3s) at six institutions[2]. It is composed of 28 major components.

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

[2]An anonymised video of Dora running can be viewed at `http://www.vimeo.com/8891653`.

Current Implementation of Framework: The framework described earlier has been implemented in Dora as an extension to the PECAS planning SA. Dora includes two goal generator components, one for each type of knowledge gap corresponding to placeholders and yet uncategorized rooms. These components monitor representations on WMs and then generate goal representations on the WM in the planning SA. As Dora is primarily concerned with filling gaps in knowledge, the importance measure proposed by the GGM framework is implemented using a heuristic measure of the *information gain* a particular goal may provide for Dora. Goals are also annotated with a heuristic value representing the estimated cost (usually in terms of time used) of achieving them. We currently do not use urgency measures as there are no meaningful time constraints in our domain. Additionally, we only employ a very limited attention filter mechanisms in order to focus on more general management principles. In accordance with the GGM framework, surfaced goals are actively managed. The goal management processes in Dora currently rank all surfaced goals according the ratio of information gain to costs, before selecting the highest-ranked goal as the next one to pursue.

In Dora, both information gain and costs are designed to reflect domain specific conditions, though their specific implementation is beyond the scope of this paper. In brief, the information gain for achieving the goal of visiting an unexplored place is derived from a measure of the amount of space predicted to be covered by this place. The information gain of categorizing a room is similarly designed, assuming that a categorising bigger rooms yields more information. The cost of exploring a place is determined by the distance to that place. The cost of categorising a room is the cost of getting there plus the predicted cost of performing AVS.

Room categorisation via AVS provides an example on how domain knowledge can be modelled in the GGM framework. Because AVS requires a map of non-free space in order to calculate a search plan, a room must be *adequately* explored before AVS can be performed in it. However, Dora does not have to *completely* explore the room (by visiting every place it contains) before this can happen. While a heuristic, dynamic definition of "adequate" can easily be implemented in a domain specific goal generator, it is more difficult to define this vague precondition in a planning domain. If we wished to remove the GGM framework from Dora (as we shall do for evaluation purposes in the subsequent section), the only natural way to encode the relationship between exploration and AVS is to add a precondition to Dora's AVS action. This precondition would have to state that no unexplored places can exist in a room if it is to be categorised, thus losing the notion of adequate exploration.

SYSTEMIC EVALUATION

To investigate the influence our GGM framework has on an integrated system, we have gathered data from multiple runs of two configurations of the Dora system. The first configuration explicitly encodes its two drives (to explore space and categorise rooms) as the single planning goal:

```
(and (forall (?p - place) (= (explored ?p) true))
     (forall (?r - room)  (kval 'robot' (areaclass ?r))))
```

This goal, literally interpreted as "explore all places and know a category for every room", is passed to the continual planner directly, rather than via the GGM framework. In this configuration, termed *conjunct goal set (CGS)*, all system behaviour is determined by the continual planner in the PECAS architecture in response to this unchanging goal. The planner continuously monitors WM content, triggering replanning if any relevant state changes occur (e.g. unexplored places appearing on the spatial WM). The second configuration, termed *managed goal set (MGS)*, employs our implementation of the GGM framework. In this configuration the planner is fed the individual goals selected by the management mechanisms described previously.

The GGM framework is an integral part of the robotic system Dora which is being run in different office environments on a regular basis. A video[3] of the real robot operating in one of these office environments can support comprehension of our evaluation setup and provides the reader with a better understanding of Dora's generated behaviour. Though Dora is usually run in such real world environments, for this study we performed the experiment in simulation for the sake of control and consistency. We used a test arena with five rooms and a corridor in the robotic simulator *Stage*[4]. The structure of the environment, which covers approximately $93m^2$, is captured in the map in Figure 2(b). Dora's sensors are simulated by Stage, allowing us to employ the same system architecture in the simulated setting as we use in the real world, with the exception of the object recogniser. This is replaced with Stage's "blobfinder". Whenever the camera is pointed towards a near-by simulated object in Stage an object is recognised. Two simulated objects are placed in each room in the arena. The objects describe one of three categories according to the OpenMind Indoor Common Sense Database (room, office, and kitchen), allowing the conceptual mapping SA to categorise these rooms.

We ran the system on a standard 2.4Ghz Core2Duo notebook with 4GB RAM. A single run is defined as follow: First, the system starts from scratch every run. Next, the robot is being given a short, predefined tour (superimposed on Figure 2(b) as a dotted arrow) during which the robot builds up its initial knowledge but it does not take any action itself. Finally, the system is switched to autonomous mode and starts acting in response to its goals
In total we ran the system 15 times: 8 in MGS configuration and 7 in CGS. This set is referred to as S_{all}. A run for the CGS configuration was defined as complete when the conjunctive goal was achieved (i.e. no places left unexplored and no rooms uncategorised). The MGS configuration was said to be complete when no more surfaced goals remained.

Results

As we are evaluating Dora under full-system conditions (rather testing using isolated components) we have to accept

[3] `http://cogx.eu/results/dora/`
[4] Available from `http://playerstage.sf.net`.

	CGS	MGS
avg. time per planning call	0.621 s	0.292 s
avg. time spent on planning	48.843 s	8.858 s

TABLE I

PLANNING TIME MEASURES (ALL IN SECONDS).

the performance limitations of the real (research) components which can occasionally fail to perform correctly. The effect of this is visible in a large variation in the time taken by Dora to complete a run (between 10 and 23 minutes). When looking at the number of rooms actually found and categorized in each run, we find a number of runs in which the robot completed the mission to categorize all five rooms. We select these runs as a subset termed S_{R5} composed of four runs of each configuration. We argue that runs in this set are comparable because the qualitative extent of the information acquired is equivalent.

To measure the influence of the GGM framework we can examine the time Dora spends planning. These measures are obtained from S_{all} and summarised in Table I. The differences between the averaged timings taken for the two configurations are statistically significant with $p < 0.0001$ in Mann-Whitney testing for all measures shown in the table.

As the first row of the table indicates, there is a significant difference between the average time taken by a single call to the planner. A call occurs either when the goal management activates a new goal or when replanning is triggered by a state change. Planning calls in CGS take more than twice the time compared tp MGS. This is due to the higher complexity of the planning problems in the CGS configuration (it is planning for all possible goals rather than a single goal). If we look at the average time spent on planning in total in a run (second row in Table I) the difference is more prominent. This is due to the fact that in the CGS configuration the planner is triggered more often: 79.0 times on average, compared to 31.1 times for the MGS configuration. This is because the longer plan lengths required in CGS are more likely to be affected by state changes and thus require more frequent replanning.

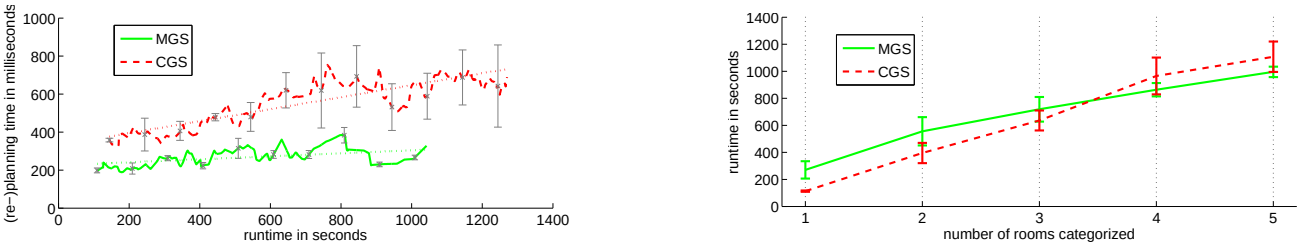
Figure 3(a) shows how the complexity of planning problems evolves as the system is running. This presents the length of single planner calls against the runtime of the system. This plot has been created using S_{R5} for comparability. We average the planning time at discrete time steps during the system's runtime, across all selected runs of each configuration. The error bars indicate the standard error in averaging. From this figure it is apparent that, in agreement with the data in Table I, less planning effort is required in MGS compared to CGS. It can also be seen that the progression over runtime is different in the two cases. While the trend, indicated by a linear fitting shown as a dotted line in Figure 3(a), is a shallowly included line for MGS, a steeper increase in average planning time can be seen for CGS. This steeper increase can be associated with the increasing size of the planning problems the CGS configuration faces as Dora's knowledge increases: planning for all possible goals over a larger and larger state becomes increasingly difficult.

The results so far underpin our hypothesis that planning effort can significantly be reduced by employing GGM mechanisms that allow a system to select the goals that should be planned for now, rather than forcing it to plan for all goals that could be possible. The reduction in effort is not a surprise, as one can argue that we are only looking at parts of the problem at a time, and thus that we are trading planning speed against the opportunity to generate plans which satisfy all the system's desires in the most efficient way possible. Contrary to this view, our experiments show that the MGS configuration satisfies its drives in a shorter time, exploring the environment more efficiently on average than the CGS configuration. So, why is this the case in our system, even though the planner considers costs by means of minimizing the number of actions to achieve the conjunctive goal? Figure 3(b) can help us understand this effect and illustrates an advantage of the MGS approach.

The plot depicts the time (y-axis) by which a given number of rooms have been categorised (x-axis), averaged over all runs of set S_{R5}. It shows that the MGS configuration takes longest to categorise the first two rooms before speeding up to be the fastest to complete four and five rooms. This observation can be explained by the use of domain-specific *information gain* for goal activation in the MGS configuration, information which is not available to the planner in either configuration. In our experiments, the MGS configuration did not choose to immediately categorise the room the tour ended in (see the arrow in Figure 2(b)). Instead it chose to visit unexplored places that were attributed with a high information gain, even though the precondition for categorising the first room was fulfilled. In contrast, the CGS configuration always categorises the first room immediately because it deems this to be the cheapest approach to achieve the overall goal. The domain knowledge encoded in the information gain used to select the goals leads to more efficient behaviour because it drives the system to explore large unexplored areas first due to their high potential information gain. This leads to the MGS configuration building up a map capable of supporting AVS on all rooms faster than the CGS configuration is able to, even if it means ignoring categorisation early on in a system run. For this reason we would expect the trend in Figure 3(b) to become more pronounced given longer runs in larger areas.

DISCUSSION & RELATED WORK

Although the problem of generating and managing goals for an integrated system has not been studied widely by the AI community, there is a body of work to which our work relates. The work of Coddington [4] supports the link between motivation and planning complexity. In a limited setting she compared two approaches to generating goals for an agent: reactive generation (comparable to our MGS configuration), and encoding all the system's goals as resources in its planning domain (comparable to our CGS configuration). This work demonstrated that by only using reactive goal generators the system could not guarantee to



(a) Averaged planning time during a system run.

(b) Running time when rooms have been categorized.

Fig. 3. Dora timing information.

satisfy all of its desires, as the effects of actions in current or future plans were not reasoned about by the generators. Coddington views using a planning process to decide which goals should be pursued as a solution to this. As a planning approach would consider all interactions between possible goals and current actions it prevents potentially deleterious situations occurring. However, Coddington demonstrated that the computational cost of encoding all possible goals in a planning problem prevented the system tackling problems beyond a certain size. Our findings, particularly those related to planning time, agree with this. Dora suffers from the problems Coddington identified with reactive goal generation, as we currently activate just one goal at a time for planning (our domain does not require more). In future work we will investigate methods for informed activation of multiple goals using oversubscription planning. This will provide a variable scale of complexity and deliberation between the two extremes of reactive and planned goal generation.

The problems and benefits of autonomous goal generation in an integrated system setting are demonstrated by the work of Schermerhorn et al. [12] present a system that can use the preconditions and effects of planning actions to generate new goals for a system at run-time. This approach has clear benefits in unpredictable worlds and would fit cleanly within a generator in our framework. [13] highlights the problems of treating goal activation as a rational decision making problem. The primary difficulty is getting reliable, comparable models of actions and the environment. Dora faces this problem when attempting to compare measures of information gain from different types of knowledge gaps.

Other frameworks for goal generation or management have been proposed previously. These approaches typically fail to make the distinction between generation, surfacing and activation, instead assuming that generation implies activation (ignoring the requirement to deliberate about possible goals). We can consider such approaches (in terms of goal generation and management) as implementations of a subset of our framework. Examples can be found in belief-desire-intention systems (e.g. [5]), behaviour-based systems (e.g. [3]), and reactive planners with goal management extensions (which represents perhaps the largest body of work on this subject) (e.g. [6], [9]).

Conclusion

In this paper we presented a framework for goal generation and management and demonstrated its efficacy in a first implementation in Dora the Explorer, an intelligent mobile robot. Our evaluation shows that a system using the framework outperforms (in terms of both planning effort and task completion time) a system in which the drives of the robot are encoded as a single planning goal. Though our current implementation is limited, we are convinced that a goal generation and management framework will allow us to work towards a principled coupling between reactive and deliberative behaviours in open-ended integrated systems in general and robotics in particular.

References

[1] L. P. Beaudoin and A. Sloman. A study of motive processing and attention. In A. Sloman, D. Hogg, G. Humphreys, A. Ramsay, and D. Partridge, editors, *Prospects for Artificial Intelligence: Proc. of AISB-93*, pages 229–238. IOS Press, Amsterdam, 1993.

[2] Michael Brenner and Bernhard Nebel. Continual planning and acting in dynamic multiagent environments. *Autonomous Agents and Multi-Agent Systems*, 19:297–331, December 2009.

[3] Joanna J. Bryson. The Behavior-Oriented Design of modular agent intelligence. In R. Kowalszyk, Jörg P. Müller, H. Tianfield, and R. Unland, editors, *Agent Technologies, Infrastructures, Tools, and Applications for e-Services*, pages 61–76. Springer, Berlin, 2003.

[4] A. M. Coddington. Integrating motivations with planning. In *Proc. of AAMAS*, pages 850–852, 2007.

[5] Michael P. Georgeff and François Felix Ingrand. Decision-making in an embedded reasoning system. In *Proc. of IJCAI*, pages 972–978, 1989.

[6] Elizabeth Gordon and Brian Logan. Managing goals and resources in dynamic environments. In Darryl N. Davis, editor, *Visions of Mind: Architectures for Cognition and Affect*, chapter 11, pages 225–253. Idea Group, 2005.

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

[8] Nick Hawes, Hendrik Zender, Kristoffer Sjöö, Michael Brenner, Geert-Jan M. Kruijff, and Patric Jensfelt. Planning and acting with an integrated sense of space. In *Proc. of Int. Workshop on Hybrid Control of Autonomous Systems*, pages 25–32, Pasadena, CA, USA, July 2009.

[9] F. Michaud, C. Côté, D. Létourneau, Y. Brosseau, J. M. Valin, É. Beaudry, C. Raïevsky, A. Ponchon, P. Moisan, P. Lepage, Y. Morin, F. Gagnon, P. Giguère, M. A. Roux, S. Caron, P. Frenette, and F. Kabanza. Spartacus attending the 2005 AAAI conference. *Auton. Robots*, 22(4):369–383, 2007.

[10] Mustafa Özuysal, Pascal Fua, and Vincent Lepetit. Fast keypoint recognition in ten lines of code. In *Proc. of CVPR*, 2007.

[11] A. Pronobis, K. Sjöö, A. Aydemir, A. N. Bishop, and P. Jensfelt. A framework for robust cognitive spatial mapping. In *Proc. of ICAR*, Munich, Germany, June 2009.

[12] Paul Schermerhorn, J Benton, Matthias Scheutz, Kartik Talamadupula, and Rao Kambhampati. Finding and exploiting goal opportunities in real-time during plan execution. In *Proc. of IROS*, St. Louis, MO, October 2009.

[13] Paul Schermerhorn and Matthias Scheutz. The utility of affect in the selection of actions and goals under real-world constraints. In *Proc. of ICAI*, 2009.

A.2 Autonomous semantic-driven indoor exploration

Bibliography H. Zender, G.J.M. Kruijff, K. Sjöö, A. Aydemir, P. Jensfelt, M. Hanheide, and N. Hawes: Autonomous semantic-driven indoor exploration, submitted to *IROS 2010*

Abstract Recent work in human-robot interaction shows how a human can guide a robot around the house on a home tour. The robot uses the interaction to include semantic information into a conceptual map, as part of the spatial model it builds up for the environment. Yet, a home tour-constructed map is typically only partial. The paper presents an approach in which a conceptual map is acquired or extended autonomously, through a closely-coupled integration of bottom-up mapping, reasoning, and active observation of the environment. The approach is novel in the non-monotonic way in which the conceptual map can be built up, and the two-way connections between perception, mapping and inference to guide semantic mapping. The approach has been fully implemented in an integrated mobile robot system. It uses OWL-based reasoning with rules and non-monotonic inference over an OpenMindderived ontology of common sense spatial knowledge, together with active visual search and information gain-driven exploration. It has been tested in different environments.

Relation to WP This paper describes the results of our systemc studies on the exploratory behaviour of our robot Dora and the non-monotinistic reasoning about the spatial composition of the explored place.

Autonomous semantic-driven indoor exploration*

H. Zender, G.J.M. Kruijff
DFKI GmbH
Saarbrücken, Germany

K. Sjöö†, A. Aydemir, P. Jensfelt
KTH
Stockholm, Sweden

M. Hanheide, N. Hawes
Un. Birmingham
Birmingham, UK

Abstract

Recent work in human-robot interaction shows how a human can guide a robot around the house on a "home tour." The robot uses the interaction to include semantic information into a conceptual map, as part of the spatial model it builds up for the environment. Yet, a home tour-constructed map is typically only partial. The paper presents an approach in which a conceptual map is acquired or extended autonomously, through a closely-coupled integration of bottom-up mapping, reasoning, and active observation of the environment. The approach is novel in the non-monotonic way in which the conceptual map can be built up, and the two-way connections between perception, mapping and inference to guide semantic mapping. The approach has been fully implemented in an integrated mobile robot system. It uses OWL-based reasoning with rules and non-monotonic inference over an OpenMind-derived ontology of common sense spatial knowledge, together with active visual search and information gain-driven exploration. It has been tested in different environments.

Introduction

Several approaches to human-augmented mapping have recently been proposed. A human guides a robot around an indoor environment, and the robot uses the information obtained through interaction with the human to semantically annotate its map. The Explorer (Zender et al. 2007), BIRON (Peltason et al. 2009), and ISAC (Kawamura et al. 2008) are just a few examples of such mobile robots.

But what happens after the home tour? After a tour, the robot typically only has a partial representation of the environment. Experience shows that human users do not necessarily visit every place, talk about every object. Even when they do, they still might be blocking the robot's view. Ontological reasoning can be used to deal with this partiality, to an extent. It can infer defaults, e.g., what objects can be found by default in a given location; cf. (Hawes et al. 2009).

*The research reported here was performed in the EU FP7 IP "CogX: Cognitive Systems that Self-Understand and Self-Extend" (ICT-215181); `http://cogx.eu`. The authors would like to thank Honda Research Institute USA Inc. for use of the OpenMind Indoor Common Sense Project data.

†K. Sjöö was supported by the Swedish Research Council, contract 621-2006-4520

The paper presents a novel approach to semantic mapping in which the robot can autonomously build a fully instantiated map. The approach presents a closely-coupled integration of several forms of cognitive functionality, in a single system. The approach combines the bottom-up construction of a conceptual map, typical for a home-tour, with autonomous exploration and top-down mechanisms for guiding visual search. Visual search, and lower levels of sensor data abstraction such as the building of topological structure, can make the mapping construction process non-monotonic. This is a natural consequence of the uncertainty and partiality of observations the robot is dealing with. Structural and conceptual abstractions may need to be reconsidered in the light of new evidence. The approach we present is capable of such non-monotonic reasoning for conceptual map construction and revision. Existing approaches for human-augmented mapping do not provide this functionality.

Below we first provide an example to illustrate the problems, and connect this to relevant background on semantic mapping. We note shortcomings, and address these in our approach. The full implementation in a mobile robot system is then presented, with a discussion of experimental results obtained in runs in several different office environments, and in simulation. We focus here on the mapping approach per se. The use of internal motivation drives and planning processes for controlling exploration is only briefly highlighted.

Example

Figure 1 illustrates the inherent non-monotonic nature of the autonomous semantic mapping process we model. (1) shows the initial state. Blue points indicate laser range readings, grey rectangles are walls, and colored circles are (linked) nodes on a navigation graph. If nodes have the same color, they are interpreted as belonging to the same room. (2) shows a sequence of nodes formed after moving around. All nodes belong to a single room, a "corridor," because the robot failed to detect the door it was passing through. In (3) the robot has passed through, and successfully detected, a doorway (red node). This triggers the creation of a new room. In (4) the robot has exited this room through another doorway, re-entering the corridor. At this point, the robot is unaware that it has returned to the same corridor as before. Only in (6) nodes become fully connected. Now, the hypothesis for a new room raised in (4) is fused with the already

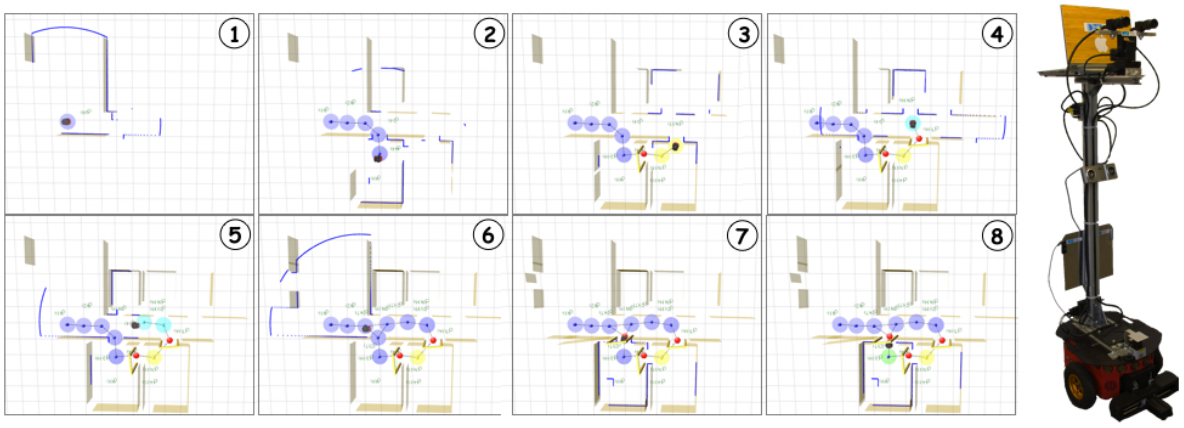


Figure 1: Actual exploration sequence. Red nodes are doorways, colored circles are free space nodes. Nodes having the same color indicates they are interpreted as belonging to the same room. Color changes of a node indicates a revision of a room hypothesis, e.g. fusion of nodes into a single room ($5 \rightarrow 6$) or separation into a new room after observing a doorway ($7 \rightarrow 8$).

existing corridor hypothesis, creating a single room. In (7), the robot detects the doorway that it had not spotted earlier, in (2). This leads to a separation of already observed nodes, creating a new room (8). The conceptual mapping approach presented here manages the potentially non-monotonic formation and maintenance of room representations. It uses topological information to establish the spatial extent of a room. Ontological inference is used to reason about the category of a room, and what objects it might contain. This in turn guides active visual search. The observations help extend the conceptual map with more instance information.

Background

There is an increased interest in including semantic information into robot-generated maps. Much of this work is driven by the need to bridge the gap between how robots and humans understand spatial organization. Humans see of spatial organization in an inherently qualitative way, using a partially hierarchical representation of topological areas.

In our approach, *Places* are the primitive topological units of the conceptual map. *Rooms* are topological units that are meaningful to humans. They are constructed from interconnected Places. Ontological reasoning determines categorical properties for a room, storing them in the robot's conceptual map. We are concerned with determining appropriate properties that allow a robot to deal with spatial entities in a way that is meaningful to a human. Properties can be inferred defaults, or categorizations of observed instances.

Several recent methods deal with constructing multi-layered environment models. Layers range from metric sensor-based maps to abstract conceptual maps that take into account information about objects acquired through computer vision methods. (Vasudevan et al. 2006) suggest a hierarchical probabilistic representation of space based on objects. (Galindo et al. 2005) presents an approach containing two parallel hierarchies, spatial and conceptual, connected through anchoring. Inference about places is based on objects found in them. The *Hybrid Spatial Semantic Hierarchy* (HSSH) of (Beeson et al. 2007) allows a mobile robot to describe the world using different representations, each with its own ontology. Compared to these our conceptual spatial representation is constructed through fusion of acquired, asserted, inferred and innate knowledge. Furthermore, our approach differs in that it provides the conceptual map as an additional abstraction layer that can be used for categorization of topological entities. Approaches like the HSSH adopt the topological layer as single abstraction.

Conceptual maps have been integrated in several mobile robots for human-robot interaction. Systems include the CoSy Explorer (Zender et al. 2007; 2008), the Cogniron BIRON system (Peltason et al. 2009; Topp et al. 2006), and ISAC (Kawamura et al. 2008). All these systems rely on the monotonic construction of a conceptual map.

Design

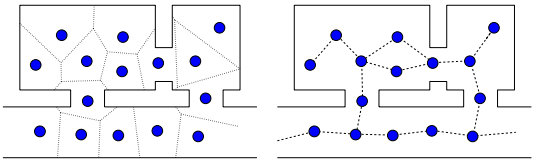
The design we present is a further development of the semantic mapping approach we developed in the CoSy Explorer system (Zender et al. 2007; 2008). That approach still assumed a strongly supervised setting, in which a conceptual map layer was built in a strictly monotonic way. Below, we present a new algorithm for managing the formation of a conceptual map layer in a way that allows for non-monotonicity. The algorithm uses the notion of topological *Places* and *Placeholders*. These are – in their turn – abstractions from metric mapping data. We first discuss the topological structure, then the conceptual mapping algorithm.

For the purposes of this paper, we only consider conceptual structures to apply to disjoint sets of Place nodes in a topological graph. As a consequence, a flat conceptual map is built, without a partonomic hierarchy based on topological inclusion. This assumption can, however, be easily lifted.

Places The robot uses laser range data to autonomously build up a 2-D metric map. This map is subdivided into dis-

crete zones called *Places*. A Place provides a basic form of spatial abstraction, cf. (Pronobis et al. 2009). Here, we define each Place in terms of a map point called a *node*. Nodes indicate "free space," and are created at regular intervals as the robot drives around the world. A node defines a Place as the Voronoi cell surrounding it, see Figure 2(a).

Nodes are connected into a *navigation graph* as the robot transits from one Place to another. Figure 2(b) illustrates such a graph. Graph-edges indicate adjacency of Places, and the possibility of moving between them. This connectivity is used in planning and conceptual reasoning.



(a) Places defined by nodes (b) Graph of Place nodes

Figure 2: Places

Placeholders Space that has not yet been explored by the robot has no Place nodes in it. Nevertheless, high-level processes like reasoning and planning do need symbols representing areas that could potentially be explored. We facilitate this by giving unexplored space its own representation in *Placeholders*. A placeholder symbolizes an unexplored direction that the robot might move in – which may or may not yield new Places. Placeholders are stored internally in the form of a position in the map termed a node *hypothesis*, generated in space that is reachable from the current Place, but which is devoid of other nearby Place nodes (Figure 3).

Placeholders and Places have the same high-level representation, as do the links connecting them. Placeholders are ascribed the additional attribute of being *unexplored*, as well as two quantitative measures of estimated information gain should the robot explore them. These are used by the motivation system, described in §*Implementation*.

The quantitative measures used are the coverage estimate and the frontier length estimate, cf. Figure 3. The former is obtained by measuring the free space that is visible from the current node and isn't near to any existing node (yellow in the figure), and assigning it to the hypothesis that is closest. This heuristically estimates the number of new Places that would result from exploring that direction.

The frontier length estimate is analogously extracted from the length of the border into unknown space. By prioritizing these two measures differently, the motivation mechanism can produce different exploratory behaviours.

Conceptual mapping Conceptual mapping uses the Place/Placeholder-based topological organization to perform two reasoning tasks. One, it maintains a representation that groups Places into rooms. Two, using observations it can infer possible categories for a room, and objects that are likely to be present by default. Performing these tasks yields a conceptual map of the environment, with room organization, instances, and default expectations.

The ongoing construction of the conceptual map is potentially non-monotonic. The overall room organization may be

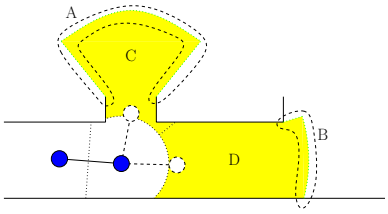


Figure 3: Placeholder creation. Dashed circles are hypotheses, each representing one placeholder. A and B are frontier length estimates, C and D are coverage estimates for the respective placeholders.

revised on the basis of new observations. The representation underlying the conceptual map is an OWL-DL ontology, consisting of a taxonomy of concepts (*TBox*) and the knowledge about individuals in the domain of discourse (*ABox*). Besides the usual inferences performed by the OWL-DL reasoner, namely *subsumption checking* for concepts in the TBox (i.e., establishing subclass/superclass relations between concepts) and *instance checking* for ABox members (i.e., inferring which concepts an individual instantiates), an additional *rule engine* is used to decide whether a pair of nodes belongs to the same room, or not, cf. (1), (2), (3). These rules are interpreted non-monotonically: whenever a previously true antecedent (left-hand side) turns false its consequent (right-hand side) statements are retracted again from the A-box. In addition, we use an algorithm responsible for room instance creation, cf. Algorithm 1.

(1) for place instances x, y:
$adjacent(x, y)$ & $\neg door(x)$ & $\neg door(y)$
$\rightarrow sameRoomAs(x, y)$

(2) for place instances x, y, z:
$adjacent(x, z)$ &$sameRoomAs(y, z)$
& $\neg door(x)$ & $\neg door(y)$ & $\neg door(z)$
$\rightarrow sameRoomAs(x, y)$

(3) for place instances x, y, and room instance z:
$sameRoomAs(x, y)$ & $contains(z, x)$
$\rightarrow contains(z, y)$

Place instances are generated in a bottom-up fashion whenever a new place node is created, or an existing placeholder turns into an explored place. Whenever the system detects a doorway at place p, the predicate $door(p)$ is added to the A-box. Likewise, if a place ceases to exist (e.g., if it gets merged with another place node), its place instance is automatically removed from the A-box. Edges between place nodes in the navigation graph are the basis for asserting their adjacency ($adjacent(x, y)$).

Rules (1) and (2) make sure that only places that are transitively interconnected (i.e., $adjacent$) without passing a doorway place are asserted to belong to the same room. The reflexive $sameRoomAs$ predicate thus provides an extensional, bottom-up definition of which segments of the navigation graph consist a room. Rule (3) makes sure that places that are in the same room (through the $sameRoomAs$ predicate), are also asserted to be contained by the same room instance (through the $contains$ predicate).

Algorithm 1 createAndDeleteRooms()

```
places ← getAllInstances(Place)
rooms ← getAllInstances(Room)
for each place ∈ places do
  if isInstanceOf(place, Door) then
    /* discard doorway places */
    places ← places − place
  else if isRelated(contains, place) then
    /* discard place if already part of a room */
    places ← places − place
  else
    /* create new room with current place as seed */
    new_room ← createNewRoomSymbol()
    addInstance(new_room, Room)
    addRelation(new_room, hasSeedPlace, place)
    addRelation(new_room, contains, place)
    contd_places ← getRelatedInstances(place, sameRoomAs)
    for each p ∈ contd_places do
      addRelation(new_room, contains, p)
      places ← places − p
      seeded_rooms ← getRelatedInstances(hasSeedPlace, place)
      for each seeded_room ∈ seeded_rooms do
        if seeded_room ≠ new_room then
          deleteInstance(seeded_room)
        end if
      end for
    end for
  end if
end for
```

Algorithm 1 handles the creation of new room instances. This is necessary as the creation of a new room symbol r cannot be expressed in the first-order logic-like rule syntax. Room creation uses the notion of *seed place*, which usually is the first node that was found to belong to a new room.

Rooms are usually extended as the robot keeps exploring.Splitting of rooms occurs when a doorway is correctly detected only later. Merging of rooms occurs when the robot enters the same room from a different side, and closes the connection to the already existing places in that room. The newer one of the two merged room instances is then deleted.

Implementation

Below we discuss how the above design has been fully implemented in a cognitive system running on a mobile platform, Figure 1. The implementation combines the spatial mapping functionality with active visual search, and motivation mechanisms to drive exploration. Motivation is planning based. It uses information gain and the current state of the map to decide whether to plan for further spatial exploration (exploring Placeholders), or for obtaining more categorical information (active visual search).

Architecture design The integrated system is built using a cognitive robotics software framework called CAST (Hawes and Wyatt 2009). CAST is an event-driven architecture, built from one or more subarchitectures. Each subarchitecture (SA) provides a certain functionality. It consists of independently executing software processing components, and a common working memory through which the components exchange information. Subarchitectures can likewise exchange information through read/write-operations on each other's working memories. We typically use robot middleware like Player/Stage to integrate sensorimotor I/O and control into a CAST system; cf. e.g. (Zender et al. 2007).

We use five subarchitectures in our system. The *spatial* SA constructs the representations of spatial knowledge. The *Active Visual Search* SA finds objects using computer vision and view planning. The *binding* SA serves to fuse information from different modalities, into singular amodal representations (Galindo et al. 2005; Jacobsson et al. 2008). The *Motivation* and *planning* SAs use the data from binding to generate goals and plans for achieving them.

Motivation SA The motivation SA is an architectural concept for *goal selection*. In the context of exploration as discussed in this paper it decides on a behavioral level, which exploration goal should be pursued next. Basically, we consider two types of goals: exploration to extend the spatial coverage of the map, or exploration to increase the amount of categorical instance information in the conceptual map.

Planning itself has been widely researched. Yet, comparatively little attention has been paid to where the goals for planning processes come from. We propose an architecture for goal generation and management based on (Wright, Sloman, and Beaudoin 1996). This architecture is composed of reactive *goal generators*, *filters*, and *management mechanisms*. The goal generators create new goals from modal content in spatial SA, and amodal content on binding SA. The filters do a first pass selection of goals to be considered for activation. Management mechanisms determine which of the remaining goals should be *activated*, i.e. planned for.

The system can generate multiple new goals asynchronously, e.g. when a new area of space is sensed, or when a command is given. At the same time it also determines which collection of goals should currently be pursued by the system, e.g. which space should be explored, or whether exploration or categorisation goals should be pursued.

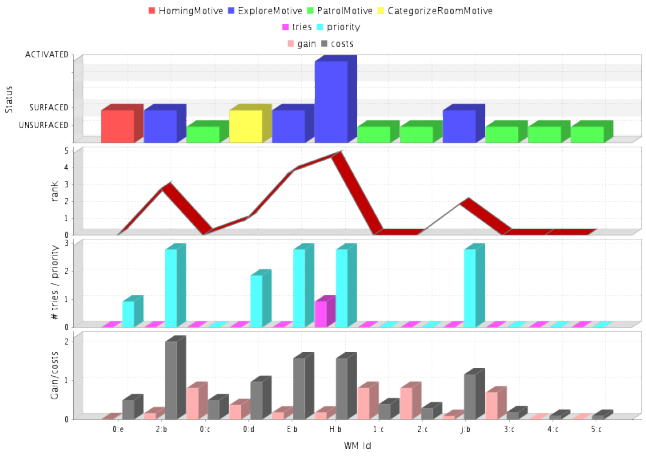


Figure 4: Screenshot of motivation SA state.

Spatial SA The SA most central to exploration is the *spatial SA*. Its components work together to extract abstract rep-

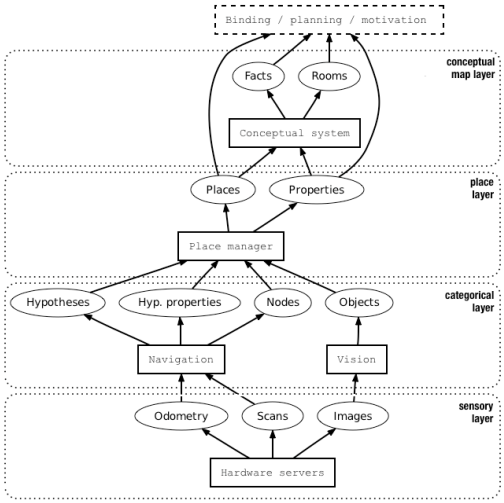


Figure 5: Data flow in the spatial SA

resentations from raw sensory data, and to translate high-level actions back to low-level motor commands. Figure 5 illustrates the data flow in the spatial SA. It is organised in a layered manner. The *sensory layer* provides continuous low-level readings from sensors. Readings are clustered and classified quantitatively in the *categorical layer*. The results are used in the *place layer* to form discrete Places and Placeholders, along with their associated properties. The components of the *conceptual map layer* perform qualitative reasoning over these abstractions.map. Firstly, the conceptual map layer segments interconnected places into rooms and maintains room instance representations, as described before. Second, the the conceptual map reasoner tries to infer more special categories for rooms, such as office or kitchen. It makes use of the inference mechanisms described in (**?**). One novelty is that the association between room categories and salient objects is done using the "locations" table of the OpenMind Indoor Common Sense database.

The output of both place and conceptual layers are presented to the system at large, through the amodal representations of the binding SA. The motivation and planning SAs use this information to decide on how to continue autonomous exploration. The motivation SA selects a goal to be pursued, e.g. for a certain placeholder to be explored, and the planning SA constructs a plan that will fulfill it. The actions that make up this plan then fed back into spatial SA, and turned into concrete continuous-space motor commands by the respective layers (not shown).

Active Visual Search SA The Active Visual Search (AVS) SA is responsible for finding visual objects in rooms. Visual search is triggered when the motivation SA selects a motive for categorizing a certain place or room. The process maintains several information flows between the AVS SA, and the spatial SA. One, observed objects are provided to the conceptual map layer in the spatial SA, to infer the category of a room using the OpenMind ontology. Two, given a motive to locate an object, the AVS SA uses information from the place- and conceptual map layers to determine in which (categorized) rooms the particular object is likely to be found in (e.g. coffee machines in kitchens).

Our implemented algorithm is a derivation of (González-Banos and Latombe 2001). Once the robot is in a room that is to be searched, the AVS SA identifies portions of the room where the object is more likely to be found. The idea behind such indirect search is that the time cost of finding possibly object-rich portions of a room is almost always smaller than a full scale random search over the whole space; cf. e.g. (Tsotsos 1992). For example, free space is assumed to be devoid of objects. At the same time, "obstacles" like landmarks that appear on the low-level map are likely to include objects. The search plan therefore starts from positions which provide the most coverage of seen obstacles, and generates view points in an art-gallery problem fashion (Shermer 1992; O'Rourke 1987).

Experiment

We tested our approach using Player/Stage, to assess the accuracy and appropriateness of our non-monotonically built spatial representation as the robot keeps exploring.

The experimental set-up was a faithful model of the real robotic platform along with its simulated sensors in a floorplan map of a part of one of our office environments, Figure 6. The map consisted of eight rooms: a corridor, a terminal room, a lab, two offices, two restrooms, and a printer room. This constitues the ground truth for our tests of the accuracy of the room maintenance. The robot was ordered to perform an autonomous exploration. The exploration was controlled by a symbolic planner and a top-level motivation system that would select appropriate locations for exploration based on the notion of placeholders. To assess the coverage that this exploration yields, we determined a gold standard of 60 place nodes to be generated to fully, densely and optimally cover the simulated environment. We achieved this by manually steering the robot to yield an optimal coverage, staying close to walls and move in narrow, parallel lanes.
We performed three runs with the robot in different starting positions, each time with an empty map. Each run was cut-off after 30 minutes. The robot was then manually controlled to take the shortest route back to its starting position.

For the evaluation, the system logged the state of its A-box each time a new room was created, or an existing one was deleted. This subsumes cases in which rooms are split or merged. At each such step, the generated map was compared to the ground truth for the room representation and to the gold standard for place node coverage. The first room instance to cover part of a ground-truth room is counted as *true positive (TP)*. If that room instance extends into a second room, it is counted as TP only once, and once as a *false positive (FP)*. Each additional room instance inside a ground-truth room is also counted as FP. *False negatives (FN)* are ground-truth rooms for which no room instance exists. Using these measures, precision, recall and the balanced f-score for the room maintenance are as follows: $precision = \#TP/\#TP + \#FP$, $recall = \#TP/\#TP + \#FN$, $fscore = 2 \times ((precision \times$

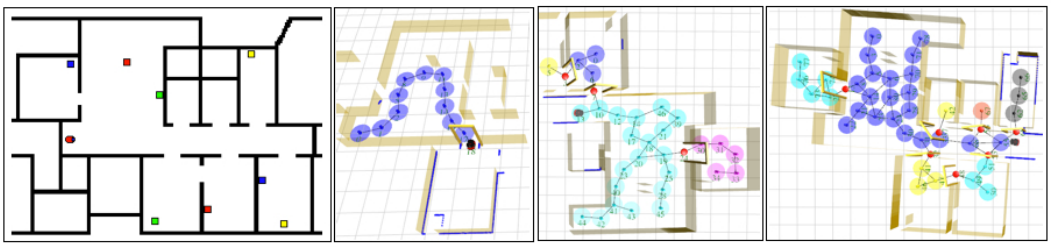


Figure 6: The environment model used in the experiments

$recall)/(precision + recall)$). We compute a normalized value for coverage using $coverage = \#nodes/60$.

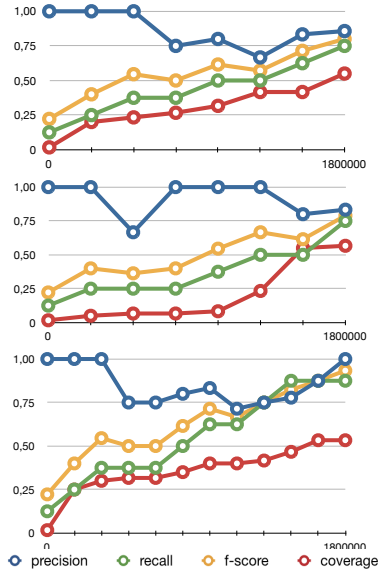


Figure 7: Plots for precision, recall, f-score and coverage of each of the three experimental runs. The Y-axis shows the normalized values for precision, recall, balanced f-score, and coverage (0–1). The X-axis is time, in milliseconds.

Figure 7 shows the development of the relevant measures during the three experimental runs. As can be seen, the accuracy (balanced f-score) of the representation is monotonically increasing. The increases and decreases in precision are due to the introduction and retraction of false room instances. Recall can be interpreted as coverage in terms of room instances. After 30 minutes the exploration algorithm yielded a relatively high recall value (3/4, 3/4 and 7/8, respectively), i.e. most of the rooms had been visited. A recurring problem here was that the two smallest rooms were often only entered by a few decimeters. This was enough to consider the corresponding placeholder to be explored, but not enough to create an additional place node beyond the doorway – which would have been the prerequisite for room instance creation. The node coverage that the algorithm achieved after 30 minutes (33, 34, 32 out of 60, respectively) can be attributed partly to the 30-minutes cut-off of the experiment, and partly to the exploration strategy which goes for high information gain placeholder first. These tend to be in the middle of a room rather than close to its walls.

Conclusions

We presented an approach that integrates several levels of cognitive functionality for a mobile robot system. The robot is able to (a) explore an indoor environment, (b) autonomously construct a multi-layered map of that environment, and (c) deliberate on the basis of the state of the map whether to explore new space, or categorize known rooms.

The approach we discussed here extends an earlier one (Zender et al. 2007). We presented a new algorithm that is capable of dealing with the partiality and uncertainty inherent to mapping. It can handle the non-monotonicity in forming and maintaining rooms. It uses OWL-DL with rule-based reasoning for room maintenance, and it is integrated with an ontology of common sense indoor environment knowledge to reason about room categories and -properties. This provides for a smooth integration with other functionality, for example for situated dialogue processing in human-robot interaction (Zender et al. 2007).

Upcoming research includes tests with the system, in longer, more complex scenarios. We are working on a setup in which the functionality described here is combined with situated dialogue processing, to develop an interactive office robot capable of finding, tracking and retrieving objects.

References

Beeson, P.; MacMahon, M.; Modayil, J.; Murarka, A.; Kuipers, B.; and Stankiewicz, B. 2007. Integrating multiple representations of spatial knowledge for mapping, navigation, and communication. In *AAAI Spring Symposium on Interaction Challenges for Intelligent Assistants*.

Galindo, C.; Saffiotti, A.; Coradeschi, S.; Buschka, P.; Fernández-Madrigal, J.; and González, J. 2005. Multi-hierarchical semantic maps for mobile robotics. In *Proc. of the IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS)*.

González-Banos, and Latombe. 2001. A randomized art-gallery algorithm for sensor placement. In *Proceedings of the seventeenth annual symposium on Computational geometry*.

Hawes, N., and Wyatt, J. 2009. Engineering intelligent information-processing systems with CAST. *Advanced Engineering Infomatics*. to appear.

Hawes, N.; Zender, H.; Sjöö, K.; Brenner, M.; Kruijff, G.-J.; and Jensfelt, P. 2009. Planning and acting with an integrated sense

of space. In *Proceedings of the 1st International Workshop on Hybrid Control of Autonomous Systems (HYCAS)*, 25–32.

Jacobsson, H.; Hawes, N.; Kruijff, G.-J.; and Wyatt, J. 2008. Crossmodal content binding in information-processing architectures. In *Proceedings of the 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*.

Kawamura, K.; Gordon, S.; Ratanaswasd, P.; Erdemir, E.; and Hall, J. 2008. Implementation of cognitive control for a humanoid robot. *International Journal of Humanoid Robotics*.

O'Rourke, J. 1987. *Art gallery theorems and algorithms*. New York, NY, USA: Oxford University Press, Inc.

Peltason, J.; Siepmann, F. H.; Spexard, T. P.; Wrede, B.; Hanheide, M.; and Topp, E. A. 2009. Mixed-initiative in human augmented mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'09)*.

Pronobis, A.; Sjöö, K.; Aydemir, A.; Bishop, A. N.; and Jensfelt, P. 2009. A framework for robust cognitive spatial mapping. In *10th Int. Conf. on Advanced Robotics (ICAR 2009)*.

Shermer, T. 1992. Recent results in art galleries [geometry]. *Proceedings of the IEEE* 80(9):1384–1399.

Topp, E.; H.Hüttenrauch; H.I.Christensen; and Eklundh, K. 2006. Bringing together human and robotic environment representations - a pilot study. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*.

Tsotsos, J. K. 1992. On the relative complexity of active vs. passive visual search. *Int. J. Comput. Vision* 7(2):127–141.

Vasudevan, S.; Gachter, S.; Berger, M.; and Siegwart, R. 2006. Cognitive maps for mobile robots an object based approach. In *Proc. of the IEEE/RSJ IROS 2006 Workshop: From Sensors to Human Spatial Concepts*.

Wright, I.; Sloman, A.; and Beaudoin, L. 1996. Towards a design-based analysis of emotional episodes. *Philosophy Psychiatry and Psychology* 3(2):101–126.

Zender, H.; Jensfelt, P.; Mozos, O. M.; Kruijff, G.; and Burgard, W. 2007. An integrated robotic system for spatial understanding and situated interaction in indoor environments. In *Proceedings of AAAI-07*, 1584–1589.

Zender, H.; Jensfelt, P.; Mozos, O. M.; Kruijff, G.; and Burgard, W. 2008. Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems* 56(6).

A.3 Dora The Explorer: A Motivated Robot

Bibliography Nick Hawes, Marc Hanheide, Kristoffer Sjöö, Alper Aydemir, Patric Jensfelt, Moritz Gbelbecker, Michael Brenner, Hendrik Zender, Pierre Lison, Ivana Kruijff-Korbayov, Geert-Jan Kruijff, and Micheal Zillich: Dora The Explorer: A Motivated Robot, accepted at *AAMAS 2010*

Abstract Dora the Explorer is mobile robot with a sense of curiosity and a drive to explore its world. Given an incomplete tour of an indoor environment, Dora is driven by internal motivations to probe the gaps in her spatial knowledge. She actively explores regions of space which she hasnt previously visited but which she expects will lead her to further unexplored space. She will also attempt to determine the categories of rooms through active visual search for functionally important objects, and through ontology-driven inference on the results of this search.

Relation to WP This short paper summarises the Dora scenario investigated in year 1. It accompanies a demo given at AAMAS 2010 where the robot is accepted to demonstrate its achievements in self-extending behaviour.

Dora The Explorer: A Motivated Robot

Nick Hawes*, Marc Hanheide
Intelligent Robotics Lab
School of Computer Science
University of Birmingham, UK

Kristoffer Sjöö, Alper Aydemir, Patric Jensfelt
Centre for Autonomous Systems
Royal Institute of Technology
Stockholm, Sweden

Moritz Göbelbecker, Michael Brenner
Institut für Informatik
Albert-Ludwigs-Universität
Freiburg, Germany

Hendrik Zender, Pierre Lison, Ivana Kruijff-Korbayová, Geert-Jan Kruijff
Language Technology Lab
German Research Center for Artificial Intelligence (DFKI)
Saarbrücken, Germany

Micheal Zillich
Vision for Robotics (V4R)
Automation and Control Institute
Vienna University of Technology

ABSTRACT

Dora the Explorer is mobile robot with a sense of curiosity and a drive to explore its world. Given an incomplete tour of an indoor environment, Dora is driven by internal motivations to probe the gaps in her spatial knowledge. She actively explores regions of space which she hasn't previously visited but which she expects will lead her to further unexplored space. She will also attempt to determine the categories of rooms through active visual search for functionally important objects, and through ontology-driven inference on the results of this search.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: Robotics

General Terms

Algorithms, Experimentation

Keywords

cognitive robotics, motivation, exploration, mapping, reasoning

1. INTRODUCTION

It has been a long standing aim of the robotics community to develop a robot capable of being a useful assistant in the home or workplace. There are a great many barriers facing such a development. One such barrier is that current systems require a lot of knowledge about an area before they can perform tasks in it. If you were to ask your interactive robot assistant "bring me the milk from the kitchen", you would only be likely to get the milk if the robot knew the complete layout of the building, how the humans working there describe the rooms, where objects are typically found, and many other things. This information could be programmed in *a priori*, or could be provided by a human during a tour when the robot was first received. These approaches have two problems. First, they are rather demanding on the time of of humans; the more information the robot requires, the more work a human has to do to provide it. This will become increasingly true as mobile vision and manipulation improves. Second, the world will continually change throughout the robot's lifetime. This will render the initial information useless, and require additional programming or human-led training.

Our solution to this problem is to allow the robot to gather knowledge autonomously. We do this by allowing it to explicitly model *gaps in its own knowledge*, which it can then proactively attempt to fill by performing knowledge gathering actions such as sensing and reasoning. This paper summarises a demo which instantiates this approach in Dora the Explorer, a mobile robot intended to perform human-specified tasks (such as the one described above) in an office environment. Dora is able to model two different types of knowledge gaps: gaps in her spatial knowledge and gaps in her knowledge about the functional categories of rooms. Spatial knowledge gaps represent areas in space which Dora knows about but hasn't visited yet. They are derived from laser scan readings combined with a metric map (built at run-time). These gaps are filled by Dora driving into the previously unvisited space. Categorical knowledge gaps represent rooms which Dora knows about, but which haven't been assigned categories. Categorical gaps are generated by ontology-based reasoning over a topological map built on top of the metric map. These gaps are filled by searching for objects in the current room and using the results to infer

*Contact author: n.a.hawes@cs.bham.ac.uk

Cite as: Dora The Explorer: A Motivated Robot, Hawes et al., *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. XXX-XXX.

its function. For example, if a stove was found in a room, Dora might hypothesise that the room is kitchen. The following section summarises the techniques used in the system to support such behaviour.

2. ARCHITECTURE

Dora's knowledge gathering is performed by following plans generated at run-time. Embedding planning into a heterogeneous robot system which itself is embodied in a dynamic, unpredictable world, requires a supporting architecture. Our architectural approach is an extension of PECAS [1]. The whole system is divided into function-based *subarchitectures*, each of which contain processing components sharing information via a working memory (WM). Modal (i.e. sensor-based) subarchitectures (e.g. mapping, vision, language) each store local representations on their WM. These modal representations are then fused into a single amodal representation by a *binding* subarchitecture, which reasons about connections between modalities. Binding provides a single view of the system's knowledge which can be used to generate planning state. The representation used by the system at this level is comparable to predicate logic. Because PECAS is intended for systems operating in multi-agent, dynamic worlds, its uses *continual planning* and *execution monitoring* to cope with partial observability and remain responsive to change.

In addition to this existing core, the Dora system incorporates a number of innovations driven by the demands of autonomous knowledge gathering: goal generation and management; planned exploration of unknown space in a new spatial model; and active visual search leading to ontology-based room categorisation. These developments, and the role they play in the demonstration, are described in the following paragraphs.

Although the process of planning has been widely researched, a comparatively small amount of attention has been directed towards where the goals for planning processes come from. In Dora we have been exploring an architecture for goal generation and management based on the work of Wright et al. [3]. This architecture is composed of reactive *goal generators* which create new goals from modal and amodal WM content; a collection of *filters* which do a first pass selection of goals to be considered for activation; and *management mechanisms* which determine which of the remaining goals should be *activated* (i.e. planned for). The architecture allows multiple new goals to be generated asynchronously by the system (e.g. when a new area of space is sensed, or when a command is given), whilst also determining which collection of goals should currently be pursued by the system (e.g. which bit of space should be explored, or which class of goal should be pursued).

A representation of space is an essential part of any mobile robot. Most current techniques provide the ability to map an area and localise within this map, but do not lend themselves to the generation of symbols for planning or other higher-level reasoning tasks. In Dora we use a new place-based representation developed with this purpose in mind [2]. In particular, Dora has been used to investigate how unexplored space can be represented in such a model. Areas where Dora's laser detects free space which is not already part of an existing place is noted as a *frontier*. Frontiers are aggregated into *placeholders* which indicate the potential for generating a new place (and thus a new spatial symbol). The presence of a placeholder triggers a goal generator to create a goal to fill the corresponding area of space by exploration. This goal is only selected if it passes through the filter and management mechanisms.

In Dora we make the assumption that the presence of particular objects determines the functional category of a room. To this end we have given Dora a decision logic-based reasoner populated with facts from the Open Mind Indoor Commonsense database describing relationships between object presence and room type (e.g. if you see a printer you might be in an office or computer room). When Dora detects a room without a category label, a goal generator creates a goal to categorise it. If this goal is activated, the plan produced causes Dora to travel to the room in question and perform a visual search for known objects. This is done by generating a view plan of regions of the room which might contain objects, then running an object recogniser from these views. When an object is found, a representation is stored on WM where the reasoner accesses it and adds it to its database. These additions, coupled with the aforementioned rules, allow Dora to infer the category of the room being searched (satisfying the planning goal).

3. DEMO

In the demo, Dora is given a short tour of an indoor area. After the tour, her goal filters are switched to allow previously generated goals to compete for activation. The user can manually adjust the filters to set priorities for classes of goals. Depending on these filters, and a cost/benefit analysis of the individual goals, Dora will select the goal or goals to pursue next, creating and executing plans to fill knowledge gaps. As she explores the world, new goals are created which enter the management architecture and influence behaviour. An example of this behaviour is that Dora can pass an open door leading to unexplored space, choose to change direction to pass through the door, then decide to explore then categorise the room beyond it. After this she can choose to readopt the goal that led her past the door originally, or choose something else that appears more rewarding. A video of the demo can be seen at http://cogx.eu/results/dora/.

4. ACKNOWLEDGMENTS

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.

5. REFERENCES

[1] N. Hawes, M. Brenner, and K. Sjöö. Planning as an architectural control mechanism. In *HRI '09: Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 229–230, New York, NY, USA, 2009. ACM.

[2] A. Pronobis, K. Sjöö, A. Aydemir, A. N. Bishop, and P. Jensfelt. A framework for robust cognitive spatial mapping. In *Proceedings of the 14th International Conference on Advanced Robotics (ICAR09)*, Munich, Germany, June 2009.

[3] I. Wright, A. Sloman, and L. Beaudoin. Towards a design-based analysis of emotional episodes. *Philosophy Psychiatry and Psychology*, 3(2):101–126, 1996.

A.4 A basic cognitive system for interactive continuous learning of visual concepts

Bibliography Danijel Skočaj, Miroslav Janíček, Matej Kristan, Geert-Jan M. Kruijff, Aleš Leonardis, Pierre Lison, Alen Vrečko, and Michael Zillich: A basic cognitive system for interactive continuous learning of visual concepts, *Subitted to ICRA 2010 workshop ICAIR - Interactive Communication for Autonomous Intelligent Robots*

Abstract Interactive continuous learning is an important characteristic of a cognitive agent that is supposed to operate and evolve in an everchanging environment. In this paper we present representations and mechanisms that are necessary for continuous learning of visual concepts in dialogue with a tutor. We present an approach for modelling beliefs stemming from multiple modalities and we show how these beliefs are created by processing visual and linguistic information and how they are used for learning. We also present a system that exploits these representations and mechanisms, and demonstrate these principles in the case of learning about object colours and basic shapes in dialogue with the tutor.

Relation to WP This paper presents the representations and mechanisms as well as the integrated system that we have developed in the George scenario. As such, it is very related to WP5 and WP6 where most of the methods were developed as well to the WP2 and WP7, where the main emphasis was on the development of the integrated system.

A basic cognitive system for interactive continuous learning of visual concepts

Danijel Skočaj, Miroslav Janíček, Matej Kristan, Geert-Jan M. Kruijff,
Aleš Leonardis, Pierre Lison, Alen Vrečko, and Michael Zillich

***Abstract*— Interactive continuous learning is an important characteristic of a cognitive agent that is supposed to operate and evolve in an everchanging environment. In this paper we present representations and mechanisms that are necessary for continuous learning of visual concepts in dialogue with a tutor. We present an approach for modelling beliefs stemming from multiple modalities and we show how these beliefs are created by processing visual and linguistic information and how they are used for learning. We also present a system that exploits these representations and mechanisms, and demonstrate these principles in the case of learning about object colours and basic shapes in dialogue with the tutor.**

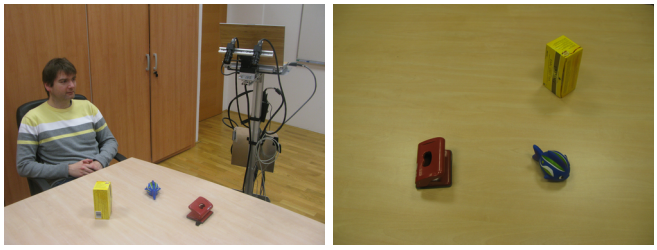
I. Introduction

An important characteristic of a cognitive system is the ability to continuously acquire new knowledge. Communication with a human tutor should significantly facilitate such incremental learning processes. In this paper we focus on representations and mechanisms that enable such interactive learning and present a system that was designed to acquire visual concepts through interaction with a human.

Such systems typically have several sources of information, vision and language being the most prominent ones. Based on the processed modal information corresponding beliefs are created that represent the robot's interpretation of the perceived environment. These beliefs rely on the particular representations of the perceived information in multiple modalities. These representations along with the cross-modal learning enable the robot to, based on interaction with the environment and people, extend its current knowledge by learning about the relationships between symbols and features that arise from the interpretation of different modalities. One modality may exploit information from another to update its current representations, or several modalities may be used together to form representations of a certain concept. We focus here on the former type of interaction between modalities and present the representations that are used for continuous learning of basic visual concepts in a dialogue with a human.

We demonstrate this approach on the robot George, which is engaged in a dialogue with the human tutor. Fig. 1 depicts a typical setup and the scene observed by the robot[1]. The main goal is to teach the robot about object properties (colours and two basic shapes). George has built-in abilities for visual processing and communication with a human, as well as learning abilities, however it does not have any model of object properties given in advance and therefore has to continuously build them. The tutor can teach the robot about object properties (e.g., 'H: This is a red thing.'), or the robot can try to learn autonomously or ask the tutor for help when necessary (e.g., 'G: Is the elongated thing red?'). Our aim is that the learning process is efficient in terms of learning progress, is not overly taxing with respect to tutor supervision and is performed in a natural, user friendly way.

In this paper we present the methodologies that enable such learning. First we present an approach for modelling multi-modal beliefs in §II. We then show how these beliefs are used in dialogue processing in §III, followed by the description of representations and the learning process in vision in §IV. In §V we describe the system we have developed and in §VI we present an example of the scenario and the processing flow. We conclude the paper with a discussion and some concluding remarks.



(a) Scenario setup. (b) Observed scene.

Fig. 1. Continuous interactive learning of visual properties.

II. Modelling beliefs

High-level cognitive capabilities like dialogue operate on high level (i.e. abstract) representations that collect information from multiple modalities. Here we present an approach that addresses (1) how these high-level representations can be reliably generated from low-level sensory data, and (2) how information arising from different modalities can be efficiently fused into unified multi-modal structures.

The approach is based on a Bayesian framework, using insights from multi-modal information fusion [1], [2]. We

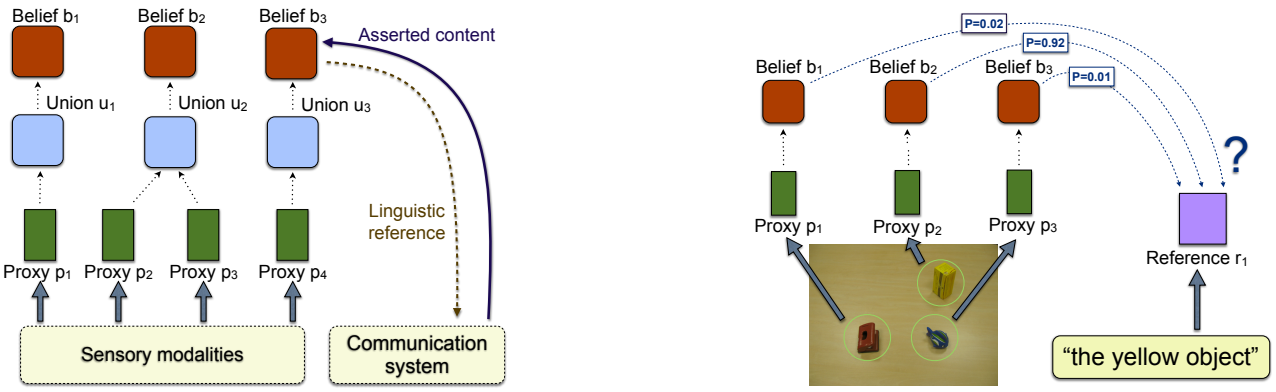
D. Skočaj, M. Kristan, A. Leonardis, and A. Vrečko are with University of Ljubljana, Slovenia, {danijel.skocaj, matej.kristan, ales.leonardis, alen.vrecko}@fri.uni-lj.si

M. Janíček, G.-J. M. Kruijff, and P. Lison are with DFKI, Saarbrücken, Germany, {miroslav.janicek, gj, plison}@dfki.de

M. Zillich is with Vienna University of Technology, Austria, zillich@acin.tuwien.ac.at

The work was supproted by the EC FP7 IST project CogX-215181.

[1]The robot can be seen in action in the video accessible at http://cogx.eu/results/george.



(a) Construction of beliefs.

(b) Reference resolution for the expression "the yellow object".

Fig. 2. Multi-modal information binding: belief construction (left) and application in a reference resolution task (right).

have implemented it as a specific subsystem called the *binder* [3]. The binder is linked to all other subsystems. It serves as a central hub for gathering information about entities currently perceived in the environment. The information on the binder is inherently probabilistic, so we can deal with varying levels of noise and uncertainty.

Based on the available information, the binder seeks to fuse the perceptual inputs arising from the various subsystems, by checking whether their respective features correlate with each other. The probability of these correlations are encoded in a Bayesian network. This Bayesian network can, for example, express a high compatibility between the haptic feature "shape: cylindrical" and the visual feature "object: mug" (since most mugs are cylindrical), but a very low compatibility between "shape: cylindrical" and "object: ball".

We call the resulting (amodal) information structure a *belief*. The task of the binder is to decide which perceptual inputs belong to the same real-world entity, and should therefore be unified into a belief. The outcome of this process is a joint probability distribution over possible beliefs. These beliefs integrate the information included in the perceptual inputs in a compact representation. They can therefore be directly used by the deliberative processes for planning, reasoning and learning.

In addition to the beliefs, there are two other central data structures manipulated by the binder, proxies and unions (see also Fig. 2(a)). A *proxy* is a uni-modal representation of a given entity in the environment. Proxies are inserted onto the binder by the various subarchitectures. They are defined as a multivariate probabilistic distribution over a set of features (discrete or continuous). A *union* is multi-modal representation of an entity, constructed by merging one or more proxies. Like proxies, unions are represented as a multivariate probabilistic distribution over possible features. They are essentially a transitional layer between proxies and beliefs.

A *belief* is an amodal representation of an entity in the environment. They are typically an abstraction over unions, expressed in an amodal format. A belief encodes additional information related to the specific situation and perspective in which the belief was formed. This includes its *spatio-temporal frame* (when and where and how an observation was made), its *epistemic status* (for which agents the belief holds, or is attributed), and a *saliency value* (a real-valued measure of the prominency of the entity [4]). Beliefs are indexed via a unique identifier, which allows us to keep track of the whole development history of a particular belief. Beliefs can also be connected with each other using relational structures of arbitrary complexity.

To create beliefs, the binder decides for each pair of proxies arising from distinct subsystems, whether they should be bound into a single union, or fork into two separate unions. The decision algorithm uses a technique from probabilistic data fusion, called the *Independent Likelihood Pool* (ILP) [5]. Using the ILP, we compute the likelihood of every possible binding of proxies, and use this estimate as a basis for constructing the beliefs. The multivariate probability distribution contained in the belief is a linear function of the feature distributions included in the proxies and the correlations between these. A Bayesian network encodes all possible feature correlations as conditional dependencies. The (normalised) product of these correlations over the complete feature set provides a useful estimate of the "internal consistency" of the constructed belief.

The beliefs, being high-level symbolic representations available for the whole cognitive architecture, provide a unified model of the environment which can be efficiently used when interacting with the human user.

III. Situated Dialogue

Situated dialogue provides one means for a robot to gain more information about the environment. A robot can discuss what it sees, and understands, with a human. Or it can ask about what it is unclear about, or would like to know more about.

That makes this kind of dialogue part of a larger activity. The human and the robot are working together. They interact to instruct, and to learn more. For that, they need to build up a common ground in understanding each other and the world.

Here we briefly discuss an approach that models dialogue as a collaborative activity. It models what is being said, and

why. It enables the robot to understand why it was told something, and what it needs to do with the information.

The approach is based on previous work by Stone & Thomason [6] (S&T). In their model, an agent uses abductive inference to construct an explanation of the possible intention behind a communicative act. This intention directs how an agent's belief models need to be updated, and what needs to be paid attention to next. This kind of inference is performed both for comprehension, and for production.

The problem with S&T is that they rely on a symmetry in communication: "What I say is how you understand it." This is untenable in human-robot interaction, particularly in a setting where a robot is learning about the world. Therefore, we have adapted and extended their approach to deal with (a) the asymmetry between what has been observed fact, and what has been asserted, and (b) clarification mechanisms, to overcome breakdowns in understanding.

Algorithm 1 Continual collaborative acting

$\Sigma^\pi = \emptyset$

loop {
Perception
  $e \leftarrow$ SENSE()
  $\langle c', i, \Pi \rangle \leftarrow$ UNDERSTAND$(r, Z(c) \oplus \Sigma^\pi, e)$
  $c \leftarrow$ VERIFIABLE-UPDATE(c', i, Π)

Determination and Deliberation
  $c' \leftarrow$ ACT-TACITLY(p, c)
  $m \leftarrow$ SELECT(p, c')
  $\langle i, \Pi \rangle \leftarrow$ GENERATE$(r, c', m, Z(c) \oplus \Sigma^\pi)$

Action
  ACT-PUBLICLY$(a(i))$
  $c \leftarrow$ VERIFIABLE-UPDATE(c', i, Π)
}

Algorithm 1 presents the core of the resulting model, based on S&T. In *perception*, the agent senses an event e. It tries to understand it in terms of an intention i that results in an update of the belief model from context c to c', given the current possible ways to do so $Z(c)$ and whatever issues are still open to be resolved Σ^π. Given the inferred intention i and potential update c' the agent then tries to carry out this update, as a *verifiable update*. To model this, we use a logical framework of multi-agent beliefs (cf. §II) that includes a notion of *assertion* [7]. An assertion is a proposition that still needs to be verified. This verification can take various forms. In George, we check whether a new piece of information can be used to consistently update a belief model (consistency), or to extend a modal model (learning) or weaken it (unlearning). Any assertion still in need of verification ends up on Σ^π.

An important aspect of linking dialogue with grounded beliefs is *reference resolution*: how to connect linguistic expressions such as "this box" or "the ball on the floor" to the corresponding beliefs about entities in the environment. The binder performs reference resolution using the same core mechanisms as used for binding. A Bayesian network specifies the correlations between the linguistic constraints of the referring expressions and the belief features (particularly, the entity saliency and associated categorical knowledge). Resolution yields a probability distribution over alternative referents (see Fig. 2(b) for an example). Abductive inference then determines which resolution hypothesis to use, in the context of establishing the best explanation. This is folded together with any new information an utterance might provide, to yield an update of the robot's current beliefs.

For example, consider an utterance like "This is yellow." First, the expression "this" must be resolved to a particular, proximal entity in the environment. Resolution is performed on the basis of the saliency measures. Second, the utterance also provides new information about the entity, namely that it is yellow. The robot's beliefs get updated with this asserted information. Dialogue processing does this by selecting the belief about the referred-to entity, then incorporating the new information. Indirectly, this acts as a trigger for learning.

In George, the dynamics of assertions on Σ^π provide the main drive for how learning and dialogue interact. The vision subarchitecture can pose *clarification requests* to the dialogue system. These requests are interpreted as tacit actions (Algorithm 1), pushing an assertion onto Σ^π. This assertion may be a polar or an open statement. Then similarly to resolving any breakdown in understanding the user, the robot can decide to generate a clarification subdialogue. This dialogue continues until the (original) assertion has been verified, i.e. a proper answer has been found [8].

IV. Learning visual concepts

In the two previous sections we discussed how the modal information gathered from individual modalities is fused into unified multi-modal structures and how they are used in situated dialogue. In this section we will describe how the modal information is captured and modelled in the visual subarchitecture; how these models are being continuously updated and how they can be queried to provide the abstracted information for higher-level cognitive processing.

To efficiently store and generalize the observed information, the visual concepts are represented as generative models. These generative models take the form of probability density functions (pdf) over the feature space, and are constructed in an online fashion from new observations. The continuous learning proceeds by extracting the visual data in the form of highdimensional features (e.g., multiple 1D features relating to shape, texture, color and intensity of the observed object) and the online Kernel Density Estimator (oKDE) [9] is used to estimate the pdf in this high-dimensional feature space. The oKDE estimates the probability density functions by a mixture of Gaussians, is able to adapt using only a single data-point at a time, automatically adjusts its complexity and does not assume specific requirements on the target distribution. A particularly important feature of the oKDE is that it allows adaptation from the positive examples (learning) as well as negative examples (unlearning) [10].

However, concepts such as *color red* reside only within a lower dimensional subspace spanned only by features that relate to color (and not texture or shape). Therefore, during the learning, this subspace has to be identified to provide the best performance. This is achieved by determining the optimal subspace for a set of mutually exclusive concepts (e.g., all colours, or all shapes). We assume that this corresponds to the subspace which minimizes the overlap of the corresponding distributions. The overlap between the distributions is measured using the multivariate Hellinger distance [9]. An example of the learnt models is shown in Fig. 3.

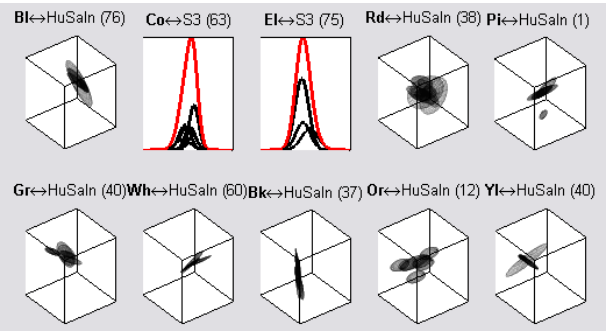


Fig. 3. Example of the models estimated using the oKDE and the feature selection algorithm. Note that some concepts are modelled by 3D distributions (e.g., "blue" which is denoted by "Bl"), while others (e.g., "compact" which is denoted by "Co") is modelled by 1D distributions.

Therefore, during online operation, a multivariate generative model is continually maintained for each of the visual concepts and for mutually exclusive sets of concepts the feature subspace is continually being determined. This feature subspace is then used to construct a Bayesian classifier for a set of mutually exclusive concepts, which can be used for recognition of individual object properties.

However, since the system is operating in an online manner, the closed-world assumption cannot be assumed; at every step the system should also take into account the probability that it has encountered a concept that has not been observed before. Therefore, when constructing the Bayesian classifier, an "unknown model" has also to be considered besides the learned models. It should account for a poor classification when none of the learnt models supports the current observation strongly enough. We assume that the probability of this event is uniformly distributed over the feature space. The a priori probability of the "unknown model" is assumed to be non-stationary and decreases with increasing numbers of observations; the more training samples the system observes, the smaller is the probability that it will encounter something novel.

Having built such a knowledge model and Bayesian classifier, the recognition is done by inspecting a posteriori probability (AP) of individual concepts and unknown model; in fact the AP distribution over the individual concepts is packed in a vision proxy, which is sent to the binder and serves as a basis for forming a belief about the observed object as described in §II (see also Fig. 2(b)).

Furthermore, such a knowledge model is also appropriate for detecting incompleteness in knowledge. It can be discovered through inspection of the AP distribution. In particular, we can distinguish two general cases. (1) In the first case the observation can be best explained by the unknown model, which indicates a gap in the knowledge; the observation should most probably be modeled with a model that has not yet been learned. A clarification request is issued that results in an open question (e.g., 'Which colour is this?'). (2) In the second case the AP of the model that best explains the observation is low, which indicates that the classification is very uncertain and that the current model cannot provide a reliable result. A clarification request is issued that results in a polar question (e.g., 'Is this red?'). In both cases, after the tutor provides the answer, the system gets the additional information, which allows it to improve the model by learning or unlearning.

V. System Architecture

We have implemented the representations and mechanisms described in the previous sections in the robot George. In this section we describe the system architecture and the individual components that are involved.

For implementation of the robot we employ a specific architecture schema, which we call CAS (CoSy Architecture Schema) [11]. The schema is essentially a distributed working memory model, where representations are linked within and across the working memories, and are updated asynchronously and in parallel. The system is therefore composed of several subarchitectures implementing different functionalities and communicating through their working memories. The George system is composed of three such subarchitectures: the *Binder SA*, the *Communications SA* and the *Visual SA*, as depicted in Fig. 4. Here, the components of the visual subsystem could be further divided into three distinct layers: the quantitative layer, the qualitative layer and the mediative layer.

In the previous subsections we assumed that the modal information is adequately captured and processed. Here we briefly describe how the relevant visual information is detected, extracted and converted in the form that is suitable for processing in the higher level processes. This is the task of *the quantitative layer* in the Visual SA. The quantitative layer processes the visual scene as a whole and implements one or more *bottom-up* visual attention mechanisms. A bottom-up attention mechanism tries to identify regions in the scene that might be interesting for further visual processing. George currently has one such mechanism, which uses the stereo 3D point cloud provided by *stereo reconstruction component* to extract the dominant planes and the things sticking out from those planes [12]. Those sticking-out parts form spherical 3D spaces of interest (SOIs). The *SOI Analyzer* component validates the SOIs and, if deemed interesting (considering SOI persistence, stability, size, etc.), upgrades them to *proto-objects* adding information that is needed for the qualitative processing, e. g. the object segmentation mask (the proto-object is segmented by the Graph cut algorithm [13] using

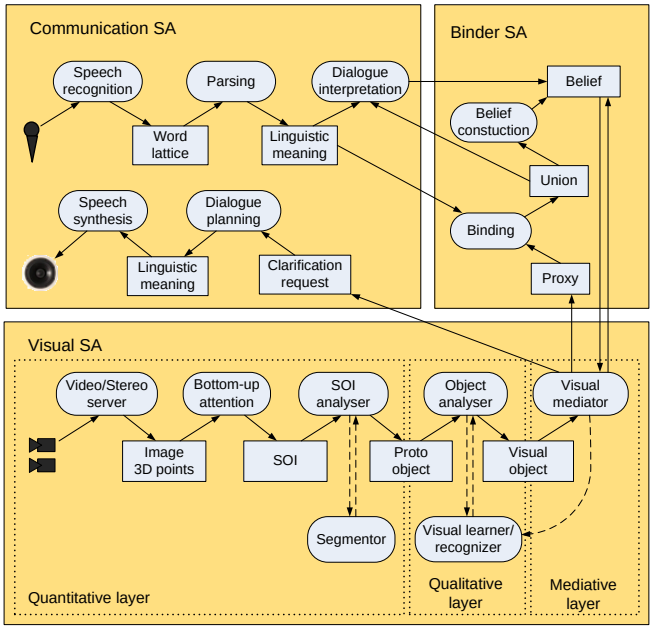


Fig. 4. Architecture of the George system.

the 3D and color information provided by the stereo reconstruction).

The qualitative layer implements the main functionalities for recognition and learning of visual concepts that were described in §IV. This layer processes each interesting scene part (object) individually, focusing on qualitative properties. After the extraction of the visual attributes (in the *Visual Learner-recognizer*), like color and shape, the *Object Analyzer* upgrades the proto-objects to *visual objects*. Visual objects encapsulate all the information available within the Visual SA and are the final modal representations of the perceived entities in the scene. Also, the learning of visual attributes is performed in this layer.

The main purpose of *the mediative layer* is to exchange information about the perceived entities with other modalities. This is not done directly, but via the specialised a-modal subarchitecture Binder SA, that actually creates and maintains beliefs as described in §II. The *Visual Mediator component* adapts and forwards the modal information about objects to the binder (each visual object is represented by a dedicated proxy in the binder). The component also monitors beliefs for possible learning opportunities, which result in modal learning actions. Another important functionality of the mediator is to formulate and forward clarification motivations in the case of missing or ambiguous modal information. Currently, these motivations are directly intercepted by the Communication SA.

Given a clarification request, the *Communication SA* formulates a dialogue goal given the information the system needs to know and how that can be related to the current dialogue and belief-context. Dialogue planning turns this goal into a meaning representation that expresses the request in context. This is then subsequently synthesized, typically as a question about a certain object property. When it comes to understanding, the Communication SA analyses an incoming audio signal and creates a set of possible word sequences for it. This is represented as a word lattice, with probabilities indicating the likelihood that a certain word was heard, in a particular sequence. The word lattice is then subsequently parsed, and from the space of possible linguistic meaning representations for the utterance, the contextually most appropriate one is chosen [14]. Finally, dialogue interpretation takes the selected linguistic meaning. This meaning is then interpreted against a belief model, to understand the intention behind the utterance. We model this is an operation on how the system's belief model is intended to be updated with the information provided. In §VI below we provide more detail, given an example.

VI. Example scenario

A. Scenario setup

The robot operates in a table-top scenario, which involves a robot and a human tutor (see Fig. 1(a)). 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. In the beginning the robot does not have any representation of object properties, therefore it fails to recognize the objects and has to learn. To begin with, the tutor guides the learning process and teaches the robot about the objects. After a while, the robot takes the initiative and tries to detect its own ignorance and to learn autonomously, or asks the tutor for assistance when necessary. The tutor can supervise the learning process and correct the robot when necessary; the robot is able to correct erroneously learned representations. The robot establishes transparency and verbalizes its knowledge and knowledge gaps. In a dialogue with the tutor, the robot keeps extending and improving the knowledge. The tutor can also ask questions about the scene, and the robot is able to answer (and keeps giving better and better answers). At the end, the representations are rich enough for the robot to accomplish the task, that is, to correctly describe the initial scene.

B. Example script

Two main types of learning are present in the George scenario, which differ on where the motivation for a learning update comes from. In tutor driven learning the learning process is initiated by the human teacher, while in tutor assisted learning, the learning step is triggered by the robot.

Tutor driven learning is suitable during the initial stages, when the robot has to be given information, which is used to reliably initiate (and extend) visual concepts. Consider a scene with a single object present:

H: Do you know what this is?
G: No.
H: This is a red object.
G: Let me see. OK.

Since in the beginning, George doesn't have any representation of visual concepts, he can't answer the question. After he gets the information, he can first initiate and later sequentially update the corresponding information.

After a number of such learning steps, the acquired models become more reliable and can be used to reference the objects. Therefore, there can be several objects in the scene, as in Fig. 1, and George can talk about them:

H: What colour is the elongated object?

G: It is yellow.

When the models are reliable enough, George can take the initiative and try to learn without being told to. In this curiosity-driven learning George can pose a question to the tutor, when he is able to detect the object in the scene, but he is not certain about his recognition. As described in §IV in such *tutor-assisted* learning there are two general cases of detection of uncertainty and knowledge gaps. If the robot cannot associate the detected object with any of the previously learned models, it considers this as a gap in its knowledge and asks the tutor to provide information:

R: Which colour is this object?

H: It is yellow.

R. OK.

The robot is now able to initialize the model for yellow and, after the robot observes a few additional yellow objects, which make the model of yellow reliable enough, it will be able to recognize the yellow colour.

In the second case, the robot is able to associate the object with a particular model, however the recognition is not very reliable. Therefore, the robot asks the tutor for clarification:

R: Is this red?

H: No. This is yellow.

R. OK.

After the robot receives the answer from the tutor, it corrects (unlearns) the representation of the concept of red and updates the representation of yellow and makes these two representations more reliable.

In such mixed initiative dialogue, George continuously improves the representations and learns reliable models of basic visual concepts. After a while George can successfully recognize the acquired concepts and provide reliable answers:

H: Do you know what this is?

G: It is a blue object.

H: What shape is the red object?

G: It is elongated.

C. Processing flow

Here we describe the processing flow for one illustrative example. We describe in more detail what happens after the human places several objects in the scene (see Fig. 1) and refers to the only elongated object in the scene (the yellow tea box) by asserting "*H: The elongated object is yellow.*".

In the Visual SA the tea box is represented by a *SOI* on the quantitative layer, a *proto-object* on the qualitative layer and a *visual object* on the mediative layer. Let us assume that the *Visual Learner-recognizer* has recognized the object as being of elongated shape, but has completely failed to recognize the color. In the binder this results in a one-proxy union with the binding features giving the highest probability to the elongated shape, while the color is considered to be unknown. This union is referenced by the single robot's private belief in the belief model (Fig. 5, step 1).

The tutor's utterance 'The elongated object is yellow' is processed by the Communication SA. Speech recognition turns the audio signal into a set of possible sequences of words, represented as a word lattice. The Communication SA parses this word lattice incrementally, constructing a representation of the utterance's most likely linguistic meaning in context [14]. We represent this meaning as a logical form, an ontologically richly sorted relational structure. Given this structure, the Communication SA establishes which meaningful parts might be referring to objects in the visual context. For each such part, the binder then computes possible matches with unions present in the binding memory, using phantom proxies (Fig. 5, step 2). These matches form a set of reference hypotheses. The actual reference resolution then takes place when we perform dialogue interpretation. In this process, we use weighted abductive inference to establish the intention behind the utterance – why something was said, and how the provided information is to be used. The proof with the lowest cost is chosen as the most likely intention. Reference resolution is done in this larger context of establishing the "best explanation." Abduction opts for that referential hypothesis which leads to the overall best proof. The resulting proof provides us with an intention, and a belief attributed to the tutor is constructed from the meaning of the utterance. In our example, this attributed belief restricts the shape to elongated, asserts the color to be yellow and references the union that includes the visual proxy representing the yellow tea box.

In the Visual SA, the mediator intercepts the event of adding the attributed belief. The color assertion and the absence of the color restriction in the robot's belief is deemed as a learning opportunity (the mediator knows that both beliefs reference the same binding union, hence the same object). The mediator translates the asserted color information to an equivalent modal color label and compiles a learning task. The learner-recognizer uses the label and the lower level visual features of the tea box to update its yellow color model. After the learning task is complete, the mediator verifies the attributed belief, which changes its epistemic status to shared (Fig. 5, step 3). The learning action retriggers the recognition. If the updated yellow color model is good enough, the color information in the binder and belief model is updated (Fig. 5, step 4).

A similar process also takes place in tutor assisted learning when the robot initiates the learning process, based on an unreliable recognition, e.g., by asking "*R: Is this red?*". In this case, the need for assistance reflects in a robot's private belief that contains the assertion about the red color and references the union representing the object. Based on this belief, the Communication SA synthesizes the above question. When the robot receives a positive answer, it updates

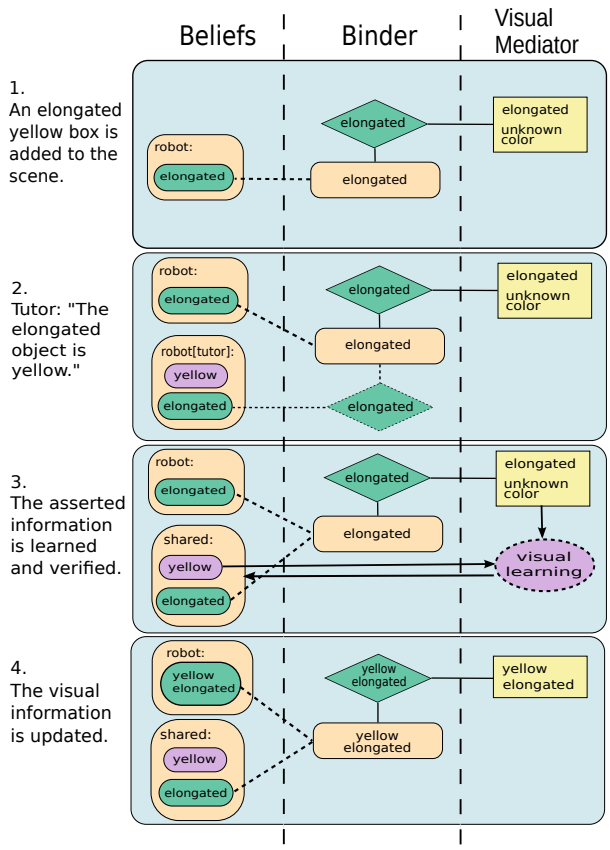


Fig. 5. Example of processing flow in the binder. The green color represents restrictive information, while the violet color denotes assertive information. Only the beliefs and other data structures pertaining to the yellow tea box are shown.

the representation of red, using a very similar mechanism as in the case of tutor driven learning.

VII. Conclusion

In this paper we presented representations and mechanisms that are necessary for continuous learning of visual concepts in dialogue with a tutor. An approach for modelling beliefs stemming from multiple modalities was presented and it was shown how these beliefs are created by processing visual and linguistic information and how they are used for learning. We also presented a system that exploits these representations and mechanisms and demonstrated these principles in the case of learning about object colours and basic shapes in a dialogue with the tutor.

We have made several contributions at the level of individual components (modelling beliefs, dialogue processing, incremental learning), as well as at the system level (by integrating the individual components in a coherent multi-modal distributed asynchronous system). Such an integrated robotic implementation enables system-wide research with all its benefits (information provided by other components), as well its problems and challenges (that do not occur in simulated or isolated environments). We are, therefore, now able to directly investigate the relations between the individual components and analyse the performance of the robot at the sub-system and system level. This will allow us to set new requirements for individual components and to adapt the components, which will result in a more advanced and robust system.

The main goal was to set up a framework that would allow the system to process, to fuse, and to use the information from different modalities in a consistent and scalable way on different levels of abstraction involving different kinds of representations. This framework has been implemented in the robot George, which is still limited in several respects; it operates in a constrained environment, the set of visual concepts that are being learned is relatively small, and the mixed initiative dialogue is not yet matured. We have been addressing these issues and the robot will gradually become more and more competent. Furthermore, we also plan to integrate other functionalities that have been under development, like motivation and manipulation.

The presented system already exhibits several properties that we would expect from a cognitive robot that is supposed to learn in interaction with a human. As such, it forms a firm basis for further development. Building on this system, our final goal is to produce an autonomous robot that will be able to efficiently learn and adapt to an everchanging world by capturing and processing cross-modal information in an interaction with the environment and other cognitive agents.

References

[1] D. Roy, "Semiotic schemas: A framework for grounding language in action and perception," *Artificial Intelligence*, vol. 167, no. 1-2, pp. 170–205, 2005.

[2] R. Engel and N. Pfleger, "Modality fusion," in *SmartKom: Foundations of Multimodal Dialogue Systems*, W. Wahlster, Ed. Berlin: Springer, 2006, pp. 223–235.

[3] H. Jacobsson, N. Hawes, G.-J. Kruijff, and J. Wyatt, "Crossmodal content binding in information-processing architectures," in *Proc. of the 3rd International Conference on Human-Robot Interaction*, 2008.

[4] J. Kelleher, "Integrating visual and linguistic salience for reference resolution," in *Proceedings of the 16th Irish conference on Artificial Intelligence and Cognitive Science (AICS-05)*, N. Creaney, Ed., 2005.

[5] E. Punskaya, "Bayesian approaches to multi-sensor data fusion," Master's thesis, Cambridge University Engineering Department, 1999.

[6] R. Thomason, M. Stone, and D. DeVault, "Enlightened update: A computational architecture for presupposition and other pragmatic phenomena," in *Presupposition Accommodation*, to appear.

[7] M. Brenner and B. Nebel, "Continual planning and acting in dynamic multiagent environments," *Journal of Autonomous Agents and Multi-agent Systems*, 2008.

[8] G. Kruijff and M. Brenner, "Phrasing questions," in *Proceedings of the AAAI 2009 Spring Symposium on Agents That Learn From Humans*, 2009.

[9] M. Kristan and A. Leonardis, "Multivariate online kernel density estimation," in *Computer Vision Winter Workshop*, 2010, pp. 77–86.

[10] M. Kristan, D. Skočaj, and A. Leonardis, "Online kernel density estimation for interactive learning," *Image and Vision Computing*, 2009.

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

[12] K. Zhou, M. Zillich, M. Vincze, A. Vrečko, and D. Skočaj, "Plane Pop-Out as 3D Attention Mechanism in a Robot Vision Domain," in *Submitted*, 2010.

[13] Y. Boykov, O. Veksler, and R. Zabih, "Efficient approximate energy minimization via graph cuts," *PAMI*, vol. 20, no. 12, pp. 1222 – 1239, 2001.

[14] P. Lison and G. Kruijff, "Efficient parsing of spoken inputs for human-robot interaction," in *Proceedings of the 18th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN 09)*, Toyama, Japan, 2009.

A.5 Plane Pop-Out as 3D Attention Mechanism in a Robot Vision Domain

Bibliography Kai Zhou, Michael Zillich, Markus Vincze, Alen Vrečko, and Danijel Skočaj: Plane Pop-Out as 3D Attention Mechanism in a Robot Vision Domain, *International Conference on Pattern Recognition 2010*

Abstract Attention operators based on 2D image cues (such as colour, texture) are well known and discussed extensively in the vision literature but are not ideally suited for robotic applications. In such contexts it is the 3D structure of scene elements that makes them interesting or not. We show how a bottom-up attention operator that selects spaces of interest (SOIs) based on scene elements that pop out from planes is used within a larger architecture for a cognitive system. In this system SOIs extracted from 3D stereo data are further refined by back-projection onto the 2D image and colour-based segmentation and finally used for tasks like learning of object properties or object recognition.

Relation to WP This paper presents the approach to bottom-up attention being integrated in the George scenario. It is key to the interactive learning scheme studied in this deliverable.

Plane Pop-Out as 3D Attention Mechanism in a Robot Vision Domain

Kai Zhou, Michael Zillich, Markus Vincze
Institute of Automation and Control, Vienna University of Technology
[zhou, zillich, vincze]@acin.tuwien.ac.at

Alen Vrečko, Danijel Skočaj
Visual Cognitive Systems Laboratory, University of Ljubljana
[alen.vrecko, danijel.skocaj]@fri.uni-lj.si

Abstract

Attention operators based on 2D image cues (such as colour, texture) are well known and discussed extensively in the vision literature but are not ideally suited for robotic applications. In such contexts it is the 3D structure of scene elements that makes them interesting or not. We show how a bottom-up attention operator that selects spaces of interest (SOIs) based on scene elements that pop out from planes is used within a larger architecture for a cognitive system. In this system SOIs extracted from 3D stereo data are further refined by back-projection onto the 2D image and colour-based segmentation and finally used for tasks like learning of object properties or object recognition.

1. Introduction

Imagine a robot entering a room with the task to locate an object, say the ubiquitous coffee mug. This is quite a challenge as the mug might be partially occluded on a cluttered office desk, hidden in shadow on a shelf, too far away and only a few pixels large or simply out of the current view. This highlights the importance of attention mechanisms for robotic vision applications, as has also been argued previously e.g. in [15]. While e.g. a lot of the object recognition literature operates on centered objects which are large in the image (at least for training) a major problem in robotic applications is to get such nice views in the first place.

Although attention has received a lot of interest in the psychology and vision literature, relatively little is concerned with attention based on 3D cues (see [4] for a good overview).

However, psychophysical studies show that spontaneous, exploratory eye movements are not only dependent on 2D features such as contrast and edge intensity as used in popular saliency models [7, 14] but are also influenced by the three-dimensional structure of the visual scene, e.g. for a slanted plane follow the depth gradient [17] or fall on the 3D center of gravity of objects rather than the 2D c.o.g. of the projection [16].

Several authors have addressed attention based on 3D cues. [10] combine disparity, image flow and motion cues into an attentional operator that is designed to follow close moving targets. [12] extend the standard Koch & Ullman [8] model of visual attention based on colour images with a depth channel. Similarly [5] combine two 2D saliency maps from the reflectance and range image of a 3D laser range scanner and show improved object recognition performance.

Putting an emphasis on biological plausibility the authors of [3] extend their Selective Tuning Model of attention to the binocular case to select areas and disparities of optimal match between left and right image. Moreover their model handles issues of binocular rivalry, i.e. can put attention on a salient region in one eye when the corresponding region in the other eye is occluded.

Showing the use of attention in a robotic system [15] present a strategy for a mobile robot equipped with a stereo head to search for a target object in an unknown 3D environment that optimises the probability of finding the target given a fixed cost limit in terms of total number of robotic actions required for detection. Their approach maintains a 3D grid of detection likelihood that is used to plan next best positions and views.

Most of the above approaches handle 3D attention by treating the disparity image like another channel next to colour. [13] use the 3D reconstructed point cloud for segmenting objects from a ground plane as a preprocessing step in a robotic scenario for learning object properties. The segmentation from stereo data, which can be of unsatisfactory accuracy depending on

the amount of available texture, is refined with graph-cut segmentation in the colour image.

We extend that approach to a wider range of scenes and develop a 3D attention operator for a robotic system aimed at various indoor tasks. Among these are object recognition and learning of objects and their properties. We make the assumption that objects presented to the robot for learning as well as objects the robot is asked to pick up are resting on supporting surfaces such as tables, shelves or simply the floor. Accordingly we place attention on anything that sticks out from supporting surfaces.

2. System Overview

Figure 2 shows (part of) the visual processing happening within a larger robotics framework. The framework is based on a software architecture toolkit [6] which handles issues like threading, lower level drivers and communication via shared working memories.

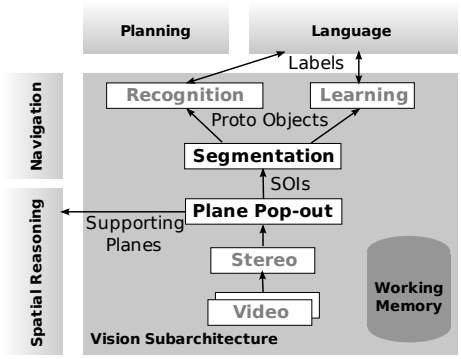


Figure 1. System overview: Attention driven visual processing in a cognitive robotics architecture

Attention in this context serves several purposes. First of all the obvious usage as a primer for costly object recognition. 3D point clouds from stereo reconstruction are used to extract dominant planes as well as things sticking out from these planes, a process which we refer to as *plane pop-out*. Those sticking-out parts form spherical 3D spaces of interest (SOIs) (termed so to avoid confusion with typical 2D image regions of interest - ROIs) which are handed to the segmentation component. The segmenter is coarsely initialised with colours obtained from back-projected 3D points inside the SOIs and then refines the projected contour of the SOIs generating what we term proto-objects. These form an intermediate level more object-like (i.e. more likely to correspond to an actual scene object) than just "stuff" that caught our attention but not quite recognised (labelled) objects yet. Proto-objects are subsequently handed to the (SIFT-based) recogniser which only needs to process the segmented image regions.

Moreover features extracted from segmented regions of interest are used to learn associations between object properties such as colour or shape and linguistic concepts such as "red" or "round".

Finally a spatial reasoning component planning where to look for objects in search tasks maintains locations of generally likely object positions, i.e. places where attention fell on in the past. To this end it stores detected supporting planes in its global map.

3. Plane Pop-Out and Segmentation

There are of course potentially many planes in an indoor environment but we are only interested in supporting, i.e. horizontal planes. We either know the tilt angle of the camera from the mounting on the mobile platform and thus know the horizontal direction or, in case we don't have a calibrated system, we find it based on the assumption that the ground plane is the dominant plane in the first view when entering a new room (initialisation phase).

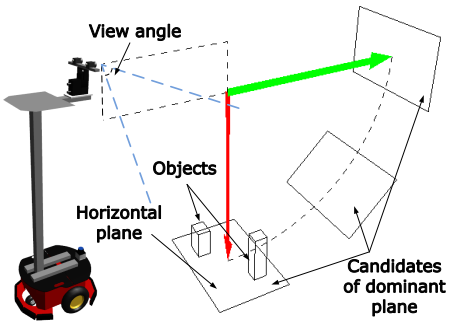


Figure 2. Illustration of hypothesis filter for initialisation: planes parallel to the stereo baseline

Plane fitting is based on RANSAC with two modifications in the hypothesis generation and in the verification stage. For the former we propose a preemptive consensus sampling scheme to increase the probability of generating valid hypotheses, i.e. horizontal planes. A sample for a plane hypothesis consists of three points and we only accept a hypothesis for further verification if the vectors between any pair of the sample points are parallel to the horizontal plane. A special case for the above initialisation phase where we do not have the horizontal plane yet is illustrated in Figure 2. Here we only require the hypothesis to be parallel to the stereo baseline, which we can always assume to be horizontal.

To improve robustness against the type of noise we have to expect in our system we dynamically adapt

the RANSAC tolerance parameter ε in the verification stage. In our case noise in the reconstructed 3D point cloud stems from disparity errors due to mismatches as well as disparity discretisation errors. In both cases the reconstruction error increases with distance from the camera. Hence we adapt ε accordingly to be more tolerant for far away hypotheses and stricter for nearer hypotheses.

We calibrate the system using two planes, the furthest (i.e. ground) and the nearest (given by the maximum disparity the stereo matching can handle). For both calibration scenes we find the smallest ε such that the best plane hypothesis contains 95% of all points and call them ε_f (far) and ε_n (near). Then with d_f, d_n and d the distances of the far, near and current hypothesis plane from the camera, ε of the current hypothesis scales linearly with distance and becomes

$$\varepsilon = \varepsilon_n + (\varepsilon_f - \varepsilon_n)\frac{d - d_n}{d_f - d_n} \tag{1}$$

Plane fitting is called iteratively until no more horizontal planes can be found. Then the remaining points sticking out from these planes are segmented using 3D flood-filling and the resulting clusters together with a bounding sphere form SOIs. Note that the bounding sphere is slightly larger than the actual point cluster to ensure it also contains a part of the plane points, which is needed for the following segmentation step. Figure 3 shows the disparity map and corresponding reconstructed point cloud for a shelf. Different planes are shown in different colours and remaining sticking out points are shown in green. Because of the inherent limitation of stereo reconstruction at poorly textured surface parts and shadowing effects between left and right camera, we refine the results using 2D colour based segmentation.

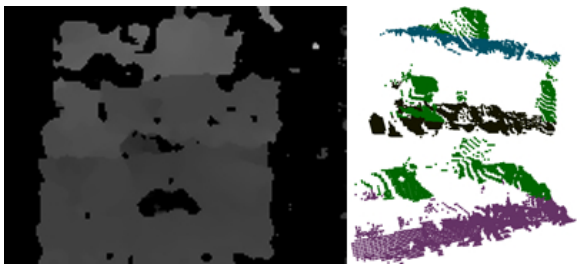


Figure 3. Disparity map and reconstructed point cloud

The 2D segmentation is based on *energy minimisation with graph cuts*. The back-projected 3D points within the SOI provide colour and spatial cues for the object and its background. The cost function for the object combines the *colour cost* with the *spatial cost*, while the cost function for the background consists of the *colour cost* component only. The *spatial cost* is simply the distance between the point and the nearest object's back-projected 3D point. The *colour cost*, on the other hand, is the average distance between the point's colour and the K nearest colours from the sample (K is determined based on the sample size). Besides foreground and background cost functions, there is a third cost function with a fixed cost to cover the areas, where both former functions have high costs. While these areas are considered uncertain and might be resolved on higher levels of the system's cognition, they are meanwhile deemed as background by the recogniser.

The distance between two colours is calculated in the HLS colour space:

$$\Delta HLS = \Delta^2 S + (1 - \Delta S)\Delta HL \tag{2}$$

$$\Delta HL = \bar{S}\Delta H + (1 - \bar{S})\Delta L, \tag{3}$$

where ΔH, ΔL and ΔS are the distances between the two colour's HLS components, while $\bar{S}$ is the average saturation of the two colours. All the parameters are normalised to values between 0 and 1. The H distance has to be additionally normalised and truncated because of its circular space. The contribution of each colour component to the overall distance between the two colours is thus determined by the saturation difference and saturation average.

The code for the graph cut algorithm was kindly provided by Boykov, Kolmogorov and Veksler [1, 2, 9].

4. Experimental Results

We tested our system on various tables and shelves. Figure 4 shows a typical result for a shelf consisting of three planes. The images show the detected ROIs (back-projected SOIs). We can see that most of the SOIs are correctly positioned on the objects sitting on the shelf except for the cluster of spray paint cans in the right corner of the second plane, which could not be reliably segmented.

Figure 4 shows the results of the subsequent segmentation step. The top images show the position of back-projected 3D points (green for object, red for background) and the segmentation (grey for object, white for background). The bottom images represent the graph cut cost functions for object and background where the brighter colour denotes greater cost. We can see that despite the fact that the backprojected 3D points are not very precise due to rather large noise, the graph-cut segmentation can be successfully initialised and provides a precise object contour.

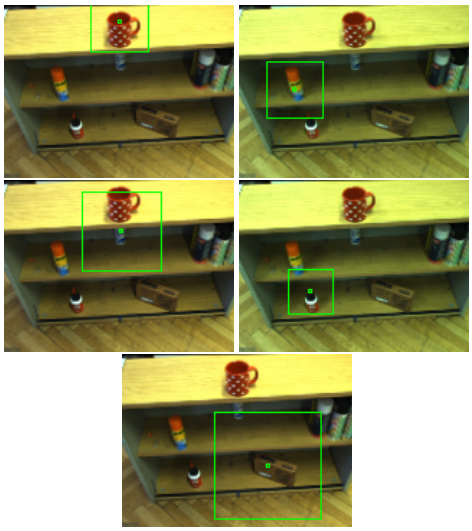


Figure 4. Spaces of interest, back-projected into image

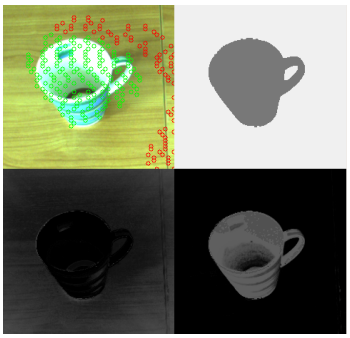


Figure 5. Segmentation of back-projected spaces of interest

5. Conclusion

We presented an attentional mechanism based on plane pop-out in 3D stereo data and its use within a robotic framework. Future work will on one hand focus on more robust extraction of supporting planes in cases where only small textured parts of the plane are visible as in the case of (densely filled) shelves. To this end we plan to fuse cues from line-based stereo with dense stereo. On the other hand we are currently integrating the recently proposed segmentation with fixation method by [11] as an alternative to the more generic graph-cut segmentation used now.

Acknowledgements

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.

References

[1] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9):1124 – 1137, 2004.

[2] Y. Boykov, O. Veksler, and R. Zabih. Efficient approximate energy minimization via graph cuts. *PAMI*, 20(12):1222 – 1239, 2001.

[3] N. D. B. Bruce and J. K. Tsotsos. An attentional framework for stereo vision. In *In Proc. of Canadian Conference on Computer and Robot Vision*, 2005.

[4] S. Frintrop, E. Rome, and H. Christensen. Computational Visual Attention Systems and their Cognitive Foundations: A Survey. *ACM Transactions on Applied Perception*, 2009.

[5] S. Frintrop, E. Rome, A. Nuchter, and H. Surmann. A bimodal laser-based attention system. *J. of Computer Vision and Image Understanding (CVIU), Special Issue on Attention and Performance in Computer Vision*, 100(1-2):124–151, 2005.

[6] N. Hawes and J. Wyatt. Engineering intelligent information-processing systems with cast. *Advanced Engineering Infomatics*, 24(1):27–39, 2010.

[7] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *PAMI*, 20(11):1254–1259, Nov 1998.

[8] C. Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. *Human Neurobiology*, pages 219–227, 1985.

[9] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2):147 – 159, 2004.

[10] A. Maki, P. Nordlund, and J.-O. Eklundh. A computational model of depth-based attention. In *Proceedings of the 13th ICPR*, volume 4, pages 734–739, Aug 1996.

[11] A. Mishra, Y. Aloimonos, and C. L. Fah. Active Segmentation with Fixation. In *ICCV*, 2009.

[12] N. Ouerhani and H. Hugli. Computing visual attention from scene depth. In *ICPR 2000*, volume 1, pages 375–378, 2000.

[13] B. Ridge, D. Skočaj, and A. Leonardis. Unsupervised learning of basic object affordances from object properties. In *Proceedings of the Fourteenth Computer Vision Winter Workshop (CVWW)*, pages 21–28, 2009.

[14] J. Tsotsos, S. Culhane, W. Wai, Y. Lai, N. Davis, and F. Nuflo. Modeling visual attention via selective tuning. *Artificial Intelligence*, 78(1–2):507–547, 1995.

[15] J. K. Tsotsos and K. Shubina. Attention and Visual Search : Active Robotic Vision Systems that Search. In *The 5th International Conference on Computer Vision Systems*, 2007.

[16] D. Vishwanath and E. Kowler. Saccadic localization in the presence of cues to three-dimensional shape. *J. Vis.*, 4(6):445–458, May 2004.

[17] M. Wexler and N. Quarti. Depth Affects Where We Look. *Current Biology*, 18:1872–1876, Dec. 2008.

A.6 Videos

- Dora the Explorer: `http://cogx.eu/results/dora`
- Curious George: `http://cogx.eu/results/george`