Indoor scene perception for object detection and manipulation

L.J. Manso, P. Bustos, P. Bachiller and J. Franco
Cceres Polytechnic School, University of Extremadura,
Cceres, Extremadura.
lmanso@unex.es

May 7, 2012

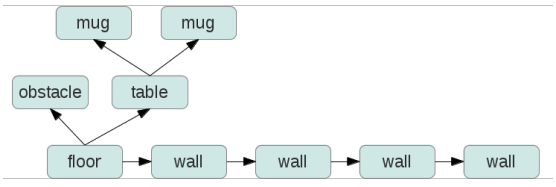
Abstract

Social robots are designed to interact and share their environments with humans while performing daily activities. They need to build and maintain rich representations of the space and objects around them in order to achieve their goals. In this paper we propose a framework for building model-based representations of the space surrounding the robots and the objects nearby. The approach considers active perception as the phenomena resulting from controlled interactions between different model-fitting algorithms and a grammar-based generative mechanism called Grammars for Active Perception (GAP). The production rules of these grammars describe how world models can be built and modified, and are associated with the behaviours needed by the model-fitting algorithms in order to succeed. Such descriptions can be used to compute the required actions to build consistent models of the environment. The resulting behaviour seizes the a priori knowledge available to the robot, not only to improve the modelling process, but also to guide exploration and visual attention. The models generated using these grammars are attributed graphs that can contain geometric and other semantic properties.

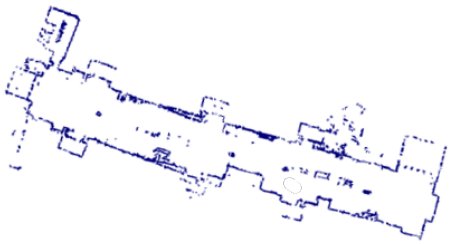
1 Introduction

Autonomous robot must be endowed with modeling capabilities in order to operate in human environments. This fact has found deep support during the last years of research in mobile robotics, where localization and mapping have captured much of the interest of the field. The mainstream approach has successfully centered on the mathematical derivation of a solution to the simultaneous localization and mapping problem (SLAM). This solution uses 3D points as the map construction material. Current research has now diversified targeting problems such as the topological organization of large point maps.

In this paper we follow a model-based approach to the spatial modeling problem that assumes there is certain amount of structured knowledge available that can be used to improve the perception process. This knowledge takes the form of simple parameterized geometric primitives (e.g., cuboids or cylinders) and compositions of them (e.g., tables or mugs) that are fitted to the sensed data through an active perception process. The uncertainty that problems such as the limited field of view of cameras, occlusions, noise or



(a) Structured high-level model (node attributes are not shown).



(b) Two-dimensional metric map in the right hand side.

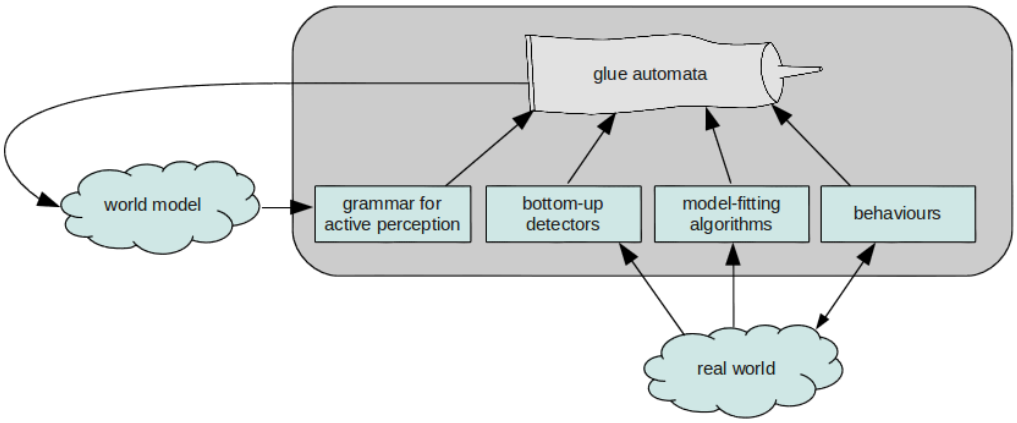


Figure 1: Organisation of the perceptive system and data flows.

limited sensor resolution introduce in the perception process can naturally be reduced by moving around and changing the point of view.

We propose a framework in which perception uses the available knowledge of the structure of the environment and enables robots to appropriately model its surroundings by taking the appropriate actions. When the robot is given a task such as please, find the blue mug and bring it here it unfolds a series of perceptive and motor processes that progressively model the key elements of its environment, constructing a safe way towards the goal. The elements needed to satisfy the goal are perceived in order, following their natural kinematic relationship (being supported by). The modelling of each new element is simplified by using a context defined by the elements already modelled. When searching for a new object, this context is subtracted from the incoming data, resulting in a crude segmentation of new potential candidates. One candidate is selected, recognized and modelled, and the cycle starts again until the plan is finished. In our example, the mug would be modelled by fitting a 3D shape on it so the arm controller can plan a grabbing behavior.

The main advantages of this type of representation are: a) that it can be used before it is completely fitted since the model is there from the beginning; b) that the model can easily adapt to environment changes once it is built and c) that the abstract nature of the models facilitates the human-robot communication. The proposed framework also provides a formal means to achieve active perception through the use of grammars. As depicted in Figure 1 it is composed of five subsystems:

- A set of bottom-up detectors that provide a shortcut among data and models. Once a model is selected, model fitting can run more efficiently.
- A set of model-fitting algorithms.
- A formal grammar that can generate the set of possible perceptive and task plans. An effective plan is opportunistically selected as time unfolds.
- A set of behaviors for low level control of the robot.
- An executive process that controls the interactions among the subsystems while moving towards the goal.

The rest of the paper describes each of these subsystems in the context of the mug task mentioned before and that requires the robot to build a representation of its environment like the one shown in Figure 1. The experiment is run in a simulator to provide an initial validation of the idea.

2 Grammars for Active Perception

Grammars are sets of rules describing how specific families of structured patterns can be incrementally built. They are generally applied to strings but they can also be applied to graph-like structures such as the one in figure 1.a. In our approach we use an extension of the concept of grammar, the so called Grammar for Active Perception (GAP). It extends previous graph-grammar formalisms to allow for unambiguous descriptions of graph transformations and, by associating behaviors to model transformations, it also provides a means for perceptive planning. GAPs can be used to reason about the valid world model transformations and about the actions required to perceive or modify specific world elements.

To design a GAP we have to define a working set of symbols. For this example task the symbols used are:

S The start symbol.

N Represents the floor normal vector of the ground plane from the point of view of the camera.

P Contains the previous data and the distance from the camera to the plane, that is, the full plane equation.

F Contains the data of the *P* symbol and the yaw angle of the robot respect to one of the four walls. It can be thought as a plane with orientation.

W, w For walls at unknown or known distance, respectively.

O For obstacles. These symbols contain their position with respect to the walls and its size.

T For tables. *T* symbols contain the position, size and height of the table.

M For mugs. The robot stores information about their color and size.

The set of rules describing how the world representation can be built is shown in Table 2. Rule 1 is triggered when the robot perceives the normal vector of the floor plane. It substitutes S with N (which has the normal vector as an attribute). Rule 2 has similar consequences but introduces the height of the camera. Rule 3 is triggered when the room orientation has been successfully modeled and also substitutes P by F, adding new information to it. The rule also includes new W symbols for the walls. Rules 4 and 5 substitute W by w when all the wall distances have been estimated. Rule 6 is triggered when the robot detects and models an obstacle. Rule 7 is used when an object previously modeled as an obstacle is found to be a table. Rule 8 is used to include new mugs in the model. As can be seen in table 2, each rule is associated with behaviors that will help the detectors in providing the necessary information to trigger the rules.

As introduced in **??**, GAPs have interesting applications. The robot achieves *bottom-up parsing* by monitoring the rules that can be potentially triggered and activating their corresponding behaviours and modeling algorithms. This also leads to *context-aware restrictions*: since only valid modeling algorithms are run, only valid transformations are introduced in the model. *Covert perception* is achieved by activating detectors of objects that can not be introduced in the model in the moment but can help detecting other objects. For example, the previous grammar can only insert mugs in tables. Thus, the table modeling process can be triggered by a table detector or be forced by the detection of a mug. *Action selection* is performed by the automata when an object detector is triggered but the robot may also choose to run a specific behaviour when it has a specific goal that has to be met.

3 Model-fitting algorithms and bottom-up detectors

Model-fitting methods are being increasingly used for scene understanding. Despite they are generally more robust to clutter and occlusion than bottom-up algorithms, they are extremely slower, especially when the object to perceive is small and has to be found first. To overcome this drawback we use both approaches in conjunction. Bottom-up detectors are used as attention-attractors to trigger model-fitting algorithms when objects have to be localised before starting their corresponding modeling process.

Monte Carlo model-fitting algorithms perform stochastic searches within the space of valid models for the most likely configuration of the parameters of the model given the input data. These approaches are generally more robust than classic bottom-up ones but the search process makes them also much slower. In order to accelerate the search we integrate bottom-up detectors to induce areas where the search

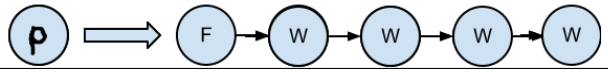
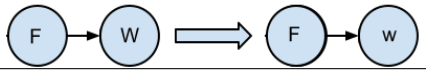
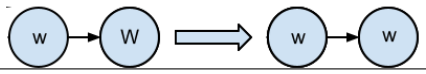
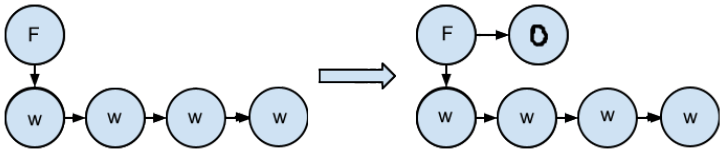
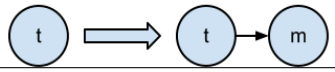
ID	Rule	Behaviour
1		watchFloor
2		watchFloor
3		alignNextWall
4		alignNextWall
5		alignNextWall
6		lookForObstacles
7		approachObstacle
8		observeTable

Table 1: Graphical description of the grammar designed for the experiment.

should take place, which is known as Data-driven Markov Chain Monte Carlo [2]. In this experiment all computations are supported by the data coming from a commercial RGBD sensor.

As indicated in section 2, the room is not modeled atomically but in different steps of two (pitch, roll), one (height) and one dimension (wall distance) respectively –the last is repeated four times for each of the walls–, which converges considerably faster than a seven-dimensional search. Moreover, it allows the robot to adopt behaviours granting appropriate points of view depending on the element to perceive (i.e., the floor in the two first cases, the walls in the last case). Each search begins with an a priori model of the room and is optimized using particle filters to fit the input data. The a priori values of the height and orientation of the camera with respect to the floor are obtained using the known kinematic structure of the robot. There is no a priori for the wall distances.

Obstacles and tables are different because their number and positions are not known. Thus, we use a discriminative detector in order to hypothesize possible candidates and trigger the corresponding behaviours and model-fitting algorithms needed to perceive tables. The detector uses the already available room model to detect clusters of points from the RGBD sensor that can not be explained by the room model (see figure **??**). When a cluster is found, the automata includes an obstacle in the model and executes an approaching behaviour to attain an appropriate point of view to run a model-fitting algorithm to fix the obstacle or transform it into a table. A similar procedure is used when detecting mugs: the mug detector subtracts the points that can be explained using the model leaving only noisy points and those that correspond to mugs. The remaining point cloud is used to fit mugs to it.

4 Behaviours

Behaviours play an essential role in robot perception. In order to perceive the environment correctly, robots have to attain favourable points of view of the parts they try to model. This must be achieved by adopting the appropriate behaviours. In the experiment the robot can adopt four behaviors. The first two are used for modeling the room:

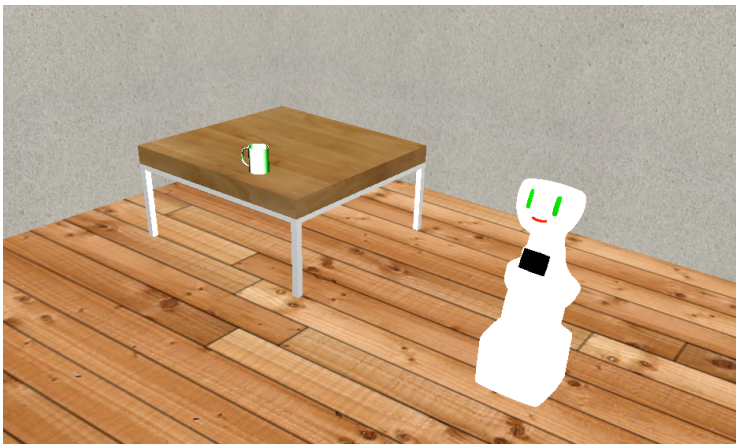


Figure 2: Environment used for the experiment.

- **watchFloor** The robot tries to localise an obstacle-free area of the floor. It is used to model the normal vector of the floor or its distance from the camera.
- **alignNextWall** Used when modeling wall-to-robot distances and when the robot selects the main orientation of the room. The robot navigates and directs its gaze towards the next wall to the right and stands still waiting for the model-fitting algorithm to finish.

The remaining two behaviors are used to model the objects in the room.

- **lookForObstacles** This behaviour makes the robot wander in order to enable the obstacle detector to do its job.
- **approachObstacle** Used to attain a favourable view of a specific obstacle and manage to fix the obstacle as it is or transform the obstacle symbol into a table, and to fit mugs to the points not explained by the model.

This small set of behaviors gives the robot the capability to model the room in which it is located, and the obstacles, tables and mugs in it. To model new objects this set should be increased correspondly as well as the grammar and model-fitting algorithms.

5 Experiment

This section describes the steps followed by the robot to achieve the goal Please, find the blue mug. The experiment is run in a simulated environment in which noise can be added to recreate real world conditions (see Figure 2).

The shortest plan is computed by the grammar engine as explained in [1] and is composed of the following sequence of rules: (1,2,3,4,5,5,5,6,7,8). In order to execute a rule the automata always performs the same tasks: a) activate the behaviour and detectors corresponding to the rule; b) activate the necessary model-fitting algorithms when the behaviour succeeds; c) trigger the rule when the model is fitted. The sequence of behaviours, detectors, model-fitting algorithms and rules triggered are shown in table 2.

6 Conclusions

We have presented a framework for active perception and the perceptive phenomena that can be achieved using it. It has been also presented its application for modeling rooms, obstacles, tables and the mugs that tables contain. The approach has been tested in a simulated robot environment with sensor noise and error control in order to recreate real conditions.

Behaviour	Detectors	Model-fitting	Rule number
watchFloor		plane normal	1
watchFloor		plane distance	2
alignNextWall		plane distance	3
watchNextWall		plane distance	4
watchNextWall		plane distance	5
watchNextWall		plane distance	5
watchNextWall		plane distance	5
lookForObstacles	obstacleDetector	obstacle sphere	6
approachObstacle		table model	7
observeTable		mug model	8

Table 2: This table shows some data

References

[1] Luis J. Manso, Pablo Bustos, Pilar Bachiller and Marco A. Gutierrez. Graph Grammars for Active Perception. *Proc. of 12th International Conference on Autonomous Robot Systems and Competitions*. Pp 63-68, 2012.

[2] Z. Tu and S-C Zhu. Image Segmentation by Data-Driven Markov Chain Monte Carlo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 24, n. 5, pp. 657-673. 2002.

[3] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J-A. Fernandez-Madrigal and J. Gonzlez. Multi-hierarchical semantic maps for mobile robotics. *International Conference on Intelligent Robots and Systems*. Pp 2278-2283, 2005.