A View-Based Navigation System for Autonomous Robots

Chandima Pathirana
Dept. of Advanced Systems Control Eng.
Graduate School of Science and Eng.
Saga University
1-Honjomachi, Saga 840-8502, Japan

Keigo Watanabe, Kiyotaka Izumi
Dept. of Advanced Systems Control Eng.
Graduate School of Science and Eng.
Saga University
1-Honjomachi, Saga 840-8502, Japan

Abstract

We present a purely vision-based scheme for learning a topological representation of an open environment. The system represents selected places by local views of the surrounding scene, and finds traversable paths between them. The set of recorded views and their connections are combined into a graph model of the environment. To navigate between views connected in the graph, we employ a homing strategy inspired by findings of insect ethology. In robot experiments, complex visual exploration and navigation tasks can thus be performed without using metric information.

1 Introduction

To survive in unpredictable and sometimes hostile environments animals have developed powerful strategies to find back to their shelter or to a previously visited food source. Successful navigation behaviour can already be achieved using simple reactive mechanisms such as association of landmarks with movements [1] or tracking of environmental features [2]. However, for complex navigation tasks extending beyond the current sensory horizon, some form of spatial representation is necessary. Higher vertebrates appear to construct representations—sometimes referred to as *cognitive maps*—which encode spatial relations between relevant locations in their environment [3, 4].

Under certain conditions, such maps can be acquired visually without any metric information. Humans, for instance, are able to navigate in unknown environments after presentation of sequences of connected views [5]. This has led to the concept of a view graph as a minimum representation required to explain experimentally observed navigation competences [7]. A view graph is defined as a topological representation consisting of local views and their spatial relations. Depending on the task, these relations can be, e.g., movement decisions connecting the views, or mere adjacencies.

Motivated by the findings of vertebrate ethology, researchers have started to investigate topological representations for robot navigation. These systems rely primarily on local sonar patterns for the identification of places, in combination with metric knowledge derived from compasses or wheel encoders. Bachelder and Waxman [8] have reported results on a vision-based topological system which uses a neural control architecture and object recognition techniques for landmark detection. In their current implementation, however, the system has to rely on artificially illuminated landmarks and a pre-programmed path during exploration of the environment. For maze-like environments, Schölkopf and Mallot [7] have shown that learning a graph of views and movement decisions is sufficient to generate various forms of navigation behavior known from rodents. The scheme has subsequently been implemented in a mobile robot [7].

The purpose of the present study is to extend the view graph approach from the mazes of [7] to open environments. In doing so, we present a navigation system that uses purely topological information based on visual input. By focusing on just one type of information we want to make the contribution of topological knowledge explicit.

2 Learning View Graphs

2.1 Discrete Representation of Continuous Space

In view-based navigation tasks, visual information is used to guide an agent through space. The reason why this is feasible at all, is the fact that there is a continuous mapping between position space (x- and y-coordinates, possibly supplemented by gaze directions) and the space of all possible views: for each spatial position, a certain view is perceived, and this view

changes continuously as the observer moves around in space. Unfortunately, this mapping can be singular, because identical views can occur at different spatial locations, i.e., there is no guarantee for the existence of a global coordinate system on the manifold of all possible views. In principle, this problem can be dealt with using context information: In points with identical views, we can use views from nearby spatial positions to disambiguate between them.

It is sufficient to store views which allow the description of relevant paths. This leads to a less detailed representation of the view manifold, namely by a graph consisting of representative views and connections between them.

Since open environments do not impose a structure on the view graph, we have to select a set of views which are representative for the manifold (in the following referred to as snapshots), and to find connections between them. Since the connecting paths between the snapshots are not explicitly coded in the view graph, we have to provide a homing method which allows us to find connected views from a given start view.

In the following sections, we introduce a system that is able to solve these tasks. The vertices of the acquired view graph are panoramic views of the environment, and their edges are connections between views that can be traversed using a visual homing procedure. This homing procedure allows the system to approach a location from any direction such that the graph edges denote mere adjacency relations without any directional labelling. The resulting view graph does not contain any explicit metric information.

2.2 A Minimal System for Learning a View Graph

The overall architecture of the system is shown in Fig. 1. Here, we discuss the basic building blocks, the details are described in the following sections.

2.2.1 *View Classifier*

As we mentioned above, a crucial component of any graph learning scheme is the selection of vertices. The graph vertices have to be distinguishable, because otherwise the representation could not be used for finding a specific location. Since we confine our system to use only visual information, we must guarantee that the recorded views are sufficiently distinct. This can be performed by a classifier which detects whether the incoming views can be distinguished from the already stored vertices of the view graph. If this condition is

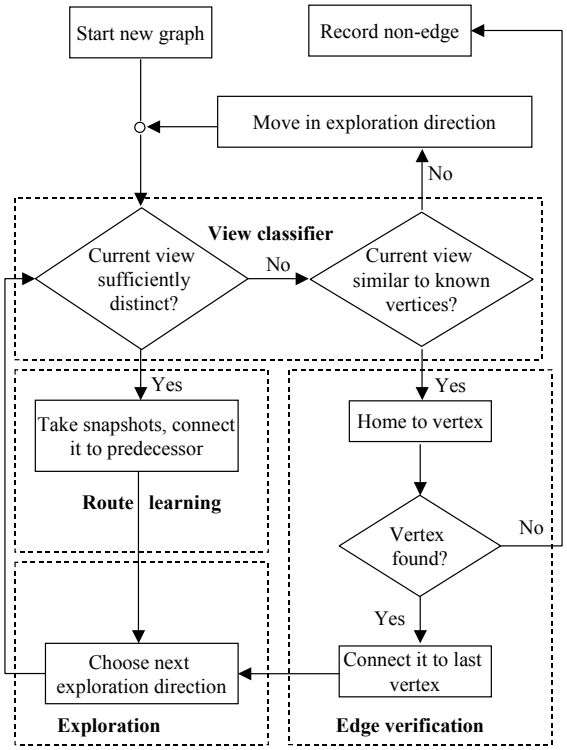

Figure 1: Block diagram of the graph learning algorithm

fulfilled, the system takes a new snapshot and adds it to the graph. The classifier is described in Section 2.3.

2.2.2 *Route Learning*

In this system, a new snapshot is automatically connected to the previously recorded vertex of the view graph. Thus, the system records chains of snapshots, or routes. These routes can be used to find a way back to the start position by homing to each intermediate snapshot in inverted order. We describe the homing procedure in Section 2.4. The area from which a specific snapshot can be reached by the homing procedure is called its catchment area. The view classifier has to make sure that every snapshot can be reached from its neighbour, i.e., all vertices of the view graph have to be in the catchment areas of their adjacent vertices.

2.2.3 *Choice of Exploration Direction*

When the system has taken a new snapshot, a new exploration direction must be chosen. This choice primarily affects the exploration strategy of this system, because it determines how thoroughly an area is explored and how fast the explored area grows. In Sec-

tion 2.5, we describe several local exploration strategies used in this system.

2.2.4 *Edge Verification*

The route learning procedure described above has no way of forming new edges to previously visited views, i.e., the resulting graphs will be mere chains. By adding the following behaviour we can obtain nontrivial graphs: during exploration, the system constantly checks whether the current view becomes similar to the already recorded snapshots. This again is a view classification task which can be solved by the same classifier as used for the selection of the snapshots (see Section 2.3). In a second step, the system checks whether the detected snapshot is not yet connected to the vertex from which the current exploration step started. Whenever these conditions hold, the system tries to home to the snapshot. If successful, an edge connecting the two vertices is included, and the exploration continues from the detected snapshot. In cases where the robot gets lost or bumps into obstacles, the system reports a "non-edge" between both vertices thus preventing the failed action from being repeated. Before starting to home, the verification procedure always checks whether a "non-edge" for this action has already been recorded. After a failed verification, we start a newgraph, which will typically get connected to the old one in due course by the edge verification procedure.

If an already connected view is encountered during an exploration step, the system homes to it as well(not shown in Fig. 1). This procedure does not produce additional knowledge, but has the effect that edges intersecting previously stored edges are less likely to be recorded.

2.2.5 *Arbitration and Obstacle Avoidance*

Since the focus of this work is on navigation, any sophisticated obstacle avoidance systems are not necessarily employed into this system. During exploration, kind of simple sensors like infrared sensors can be used for the presence of nearby objects. If obstacles were detected at distances larger than 1 cm, the robot is made to turn away without slowing down. Smaller distances can be interpreted as collisions causing the robot to back up and turn away from the obstacle. Both behaviors and the graph learning system of Fig. 1 are combined into a subsumption architecture where the collision-induced escape behavior had highest, the graph learning procedure lowest priority.

The robot is not allowed to take snapshots as long as the obstacle avoidance system is active. The resulting graph structure tends to concentrate in the open space between obstacles. This feature makes the navigation system more effective, because the visual input changes very rapidly near objects. Exploration of these areas would require a large number of snapshots which, in complex natural environments, would ultimately lead to a fractal graph structure near objects.

2.3 View Classifier

Ideally, the set of snapshots taken to represent the view manifold should satisfy three criteria: first, the views should be distinguishable. In purely graph-based maps, this is the only way to guarantee that specific vertices can be navigated to. This can be achieved by incorporating only distinct views into the graph. Second, a large proportion of the view manifold should be covered with a small number of vertices to keep processing requirements small. Third, the spatial distance of neighbouring views should be small enough to allow reliable homing between them.

As we confine this system to use only visual input, the selection of the snapshots must be based on the current view and the stored snapshots. The above criteria can be satisfied by measuring the degree of similarity between views: dissimilar views are distinguishable by definition while being distant on the view manifold, and similar views often are spatially close.

Clearly, a threshold of classifier can also be used to detect whether the current view becomes similar to one of the already recorded snapshots. If the image distance to a snapshot falls below the threshold, the robot starts its edge verification procedure (as stated in Section 2.2) and tries to home to the snapshot. In this system, we use the same classifier for both tasks. A suitable threshold can be determined experimentally.

2.4 Navigating Between Places: View-Based Homing

In order to travel between the vertices of the view-graph, we need a visual homing method. Since the location of a vertex is only encoded in the recorded view, we have to deduce the driving direction from a comparison of the current view to the goal view. After the robot has moved away from the goal, the images of the surrounding landmarks in the current view are displaced from their former image positions in the goal view. If the robot moves so as to reduce these displacements, it will finally find back to the goal where current view and snapshot match. The displacement

field has a focus of contraction in the goal direction. Driving into the direction of this focus most quickly reduces the image displacements.

A number of experiments have shown that invertebrates such as bees or ants are able to pinpoint a location defined by an array of nearby landmarks. Apparently, these insects search for their goal at places where the retinal image forms the best match to a memorized snapshot. Cartwright and Collett [9] have put forward the hypothesis that bees might be able to actively extract the goal direction by a homing mechanism as described above.

2.5 Local Exploration Strategies for Graph Learning

The exploration strategies used by this system have been motivated by the principle of maximizing knowledge gain [6]. As we have not formalized any notion of knowledge, this principle was used as a qualitative guideline. In this context, knowledge gain is possible, for instance, through the recording of new edges and new snapshots. In the following, we describe several exploration strategies, which concern primarily the choice of the next direction to explore after a snapshot has been taken, or after an existing vertex has been reached (as discussed in Section 2.2).

2.5.1 *Exploration Direction During Route Learning*

The simplest conceivable rule is to choose a random direction and then to go straight until the next snapshot is taken. The resulting Brownian motion pattern has the advantage that eventually every accessible point of the environment will be explored without the danger that the exploring agent is caught in an infinite loop. Good results can also be achieved if one uses a fixed turning angle. Using smaller angles, distant areas are reached faster, whereas angles closer to lead to a more thorough exploration of the local neighbourhood.

2.5.2 *Exploration of the Largest Open Angle*

This navigation scheme is designed such that all vertices of the view graph remain in the catchment areas of their respective neighbours. This property can be used to choose the next exploration direction, if a vertex has already more than one edge: the system determines the directions to all neighbouring vertices using the homing procedure, and directs the next exploration step to the largest open angle between them. Alternatively, one could use information about neighbouring vertices, such as their connectivity or similarity. For example, exploring areas where neighbouring views are connected to each other would be more likely to lead to possibly undesired edge intersections.

3 Summary

This paper has presented the view graph learning method that can be used for view based navigation system of autonomous robots. This method is expected to incorporate into a real robotic system which is expected to work only on visual inputs.

References

[1] R. Wehner, B. Michel, and P. Antonsen, "Visual navigation in insects: Coupling of egocentric and geocentric information," *J. Exp. Biol.*, vol. 199, pp. 129–140, 1996.

[2] T. S. Collett, "Insect navigation en route to the goal: Multiple strategies for the use of landmarks," *J. Exp. Biol.*, vol. 199, pp. 227–235, 1996.

[3] J. O'Keefe and L. Nadel, *The Hippocampus as a Cognitive Map*, Clarendon Press: Oxford, 1978.

[4] R. Gallistel, *The Organization of Learning*, MIT Press: Cambridge, MA, 1990.

[5] M. J . O'Neill, "Evaluation of a conceptual model of architectural legibility," *Environment and Behavior*, vol. 23, pp. 259–284, 1991.

[6] S. Thrun, "Exploration in active learning," in *The Handbook of Brain Theory and Neural Networks*, M.A.Arbib(Ed.), MIT Press, pp. 381–384, 1995.

[7] B. Schölkopf and H. A. Mallot, "View-based cognitive mapping and path planning," *Adaptive Behavior*, vol. 3, pp. 311–348, 1995.

[8] I. A. Bachelder and A. M. Waxman, "A view-based neurocomputational system for relational map-making and navigation in visual environments," *Robotics and Autonomous Systems*, vol. 16, pp. 267–289, 1995.

[9] B. A. Cartwright and T. S. Collett, "Landmark learning in bees," *J. Comp. Physiol. A*, vol. 151, pp. 521–543, 1983.