Topological Navigation in Configuration Space Applied to Soccer Robots

Gonçalo Neto, Hugo Costelha, and Pedro Lima

Instituto de Sistemas e Robótica
Instituto Superior Técnico
Av. Rovisco Pais, 1 – 1049-001 Lisboa, Portugal
{gneto,hcostelha,pal}@isr.ist.utl.pt
http://socrob.isr.ist.utl.pt

Abstract. This paper describes a topological navigation system, based on the description of key-places by a reduced number of parameters that represent images associated to specific locations in configuration space, and the application of the developed system to robotic soccer, through the implementation of the developed algorithms to RoboCup Middle-Size League (MSL) robots, under the scope of the SocRob project (*Soccer Robots* or *Society of Robots*). A topological map is associated with a graph, where each node corresponds to a key-place. Using this approach, navigation is reduced to a graph path search. Principal Components Analysis was used to represent key-places from pre-acquired images and to recognize them at navigation time. The method revealed a promising performance navigating between key-places and proved to adapt to different graphs. Furthermore, it leads to a robot programming language based on qualitative descriptions of the target locations in configuration space (e.g., Near Blue Goal with the Goal on its Left). Simulation results of the method application are presented, using a realistic simulator.

1 Introduction

The problem of robot navigation is, perhaps, one of the key issues in mobile robotics. Roughly, it consists in driving a robot through a given environment, using the information from his sensors.

The most common form of solving this problem is to construct a world model from the sensorial information. Based on this model, it is relatively easy to apply control algorithms and drive the robot to its target locations. One of the problems with this approach is the amount of computational effort required to store the above-mentioned world model. Even when that is not an issue, most of the times the sensorial information is not as exact as it would be desirable.

In this line of thought, approaching the problem in a more qualitative manner seems to be quite a promising alternative, in which the relevant places of the world are determined from their appearance. Vision sensors are normally used to extract the relevant information, as opposing to non-vision sensors which are widely used in the construction of world models. The problem of appearance

D. Polani et al. (Eds.): RoboCup 2003, LNAI 3020, pp. 551–558, 2004.

based methods for navigation is surely not a closed one and has been addressed in various manners, in a recent past.

In [1] the localization of the robot in a topological map is obtained by analyzing images in the frequency domain. On the other hand, [2] uses the original input images and [3] uses the color histograms of such images. An interesting idea is to use Principal Component Analysis methods to extract the most important characteristics from an image or group of images, as proposed in [4].

This paper introduces the application of topological navigation methods to robotic soccer. We have developed a flexible method that allows the robot to pursue its navigation objectives, during a match, using a qualitative approach, making the interaction with higher knowledge levels natural. The methodology used in solving the overall problem is summarized in Fig. 1.

The outline of this paper is as follows: Section 2 describes the PCA methods used to construct the topological map. Section 3 presents the techniques used for localization and navigation. In Section 4 the application of the previous methods to robotic soccer is explained and Section 5 presents the main results. Finally in Section 6 some conclusions are drawn and future research perspectives explained.

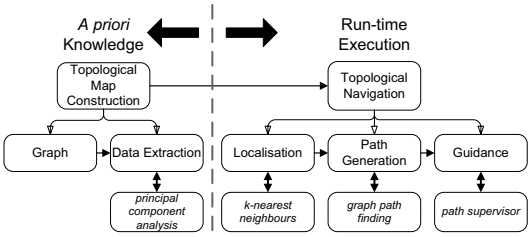

Fig. 1. Summary of methods used in this work

2 Map Construction

The first step of the topological navigation approach is to build a representation for the topological map, based on which the robot will navigate. Several representations of this map can be used; in our case we have chosen a directed graph, where the nodes represent the key-places in the map and the transitions represent the functions used to navigate between key-places.

This kind of representation is general enough to be used in a large number of applications. In our case, the nodes will be identified with groups of postures in the configuration space (x, y, θ) and the transitions correspond to basic functions like *move back* or *move forward*.

After defining the topological map, the information required to represent each of the nodes must be gathered. As this is an appearance based method, we first start to acquire a set of images P that represents the space where the robot will navigate. This set of images has to be general enough to represent all the areas of the configuration space where we want the robot to navigate.

The following step is to use Principal Component Analysis (PCA), also known as Karhunen-Loeve (KL) expansion [5, 6], to compress the information in P. At the end of the expansion, a basis for the η-dimensional that best approximates the acquired images, in the least squares sense, is obtained. This method is equivalent to retain the directions with the larger variance (i.e., the largest amount of information) of the data set.

Furthermore, it's possible to define the error we are making in the approximation, by the following expression:

$$E[\varepsilon^2] = \sum_{i=\eta+1}^{M} \lambda_i \tag{1}$$

or, in percentage terms:

$$\xi = \frac{\sum_{i=\eta+1}^{M} \lambda_i}{\sum_{i=1}^{M} \lambda_i} \tag{2}$$

where η corresponds to the number of eigenvalues chosen to represent the eigenspace. These expressions provide a criterion to choose the number of eigenvectors to be used.

Having explained the idea behind the approximation, an iterative procedure can be used to compute the principal images, as explained in [5]. We call *principal space* to the space obtained after all these calculations.

The next step in the construction of the topological map is to associate the previously gathered information with the nodes of the graph. In this part of the method, we start by projecting each image in the principal space, associating the projection with the node of the graph that the corresponding image represents. Of course, it is essential that the input images are previously grouped according to the node they best characterize.

We can think of the obtained groups of projections as classes of patterns, which allows the localization problem to be formulated as a pattern classification one.

3 Topological Navigation

With the tools to represent the topological map available, it is possible to define the methods to use in the navigation itself. So, we start by presenting the localization problem and then move to the discussion of the path planning and path execution problems.

It must be pointed out that navigation procedures, as opposed to map construction, must be executed in real-time.

As previously stated, the localization of a certain image in a node of the graph can be formulated as a classification problem. To solve this problem, a known classifier can be used. We chose to classify the images in the *k-nearest neighbor* [7] sense. In this classifier, the current input image projection is compared to the projections of all the images used in the map construction (also known as

learning) stage of the method. We then retain the k closest images and classify the image in the most represented class among the selected k images.

A path to the objective node of the robot was obtained using widespread search algorithms [8]. Of course, an heuristic could be used to improve the search efficiency.

The other step in the navigation process concerns guidance, in which we must ensure that the robot accomplishes the goals set by the path planner. In this context, it is straightforward to achieve such goals, since the path is defined as a sequence of primitive transition functions. The guidance loop for this approach is described through the flowchart in Fig. 2. The problem with this solution is

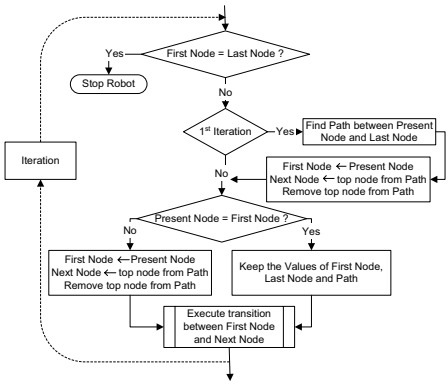

Fig. 2. Open loop guidance supervisor

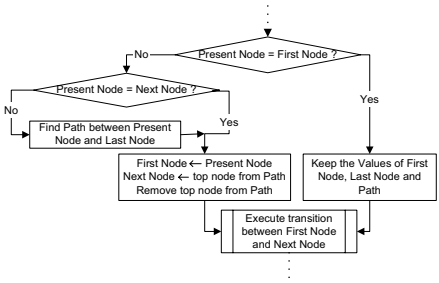

Fig. 3. Modified open loop guidance supervisor, to include path replanning

that, if a transition fails and the robot finds itself, e.g., in a node which does not belong to the path, it simply cannot fulfill its objectives. A straightforward solution for this problem is to generate a new path each time the robot detects a failure in a transition, as explained in the flowchart of Fig. 3.

4 Application to Robotic Soccer

Our team of robots, the RoboCup MSL *ISocRob* team, consists of four *Nomadic Super-Scout II* robots, which have, among other sensors, a front camera and moves based on a differential drive kinematic structure.

In order to better test the previously described navigation method, we developed a simulator, which could generate images similar to the front camera of our robots. This simulator was implemented in *Virtual Reality Modeling Language* (VRML), and field textures were used to improve the realism. The images obtained were RGB images with 320×240 pixels taken over a field with $10 \times 5[m]$.

Figure 4 compares a simulator image and the image seen by the robot's front camera, in a real situation.

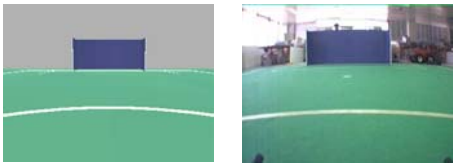

Fig. 4. Simulated vs Real images

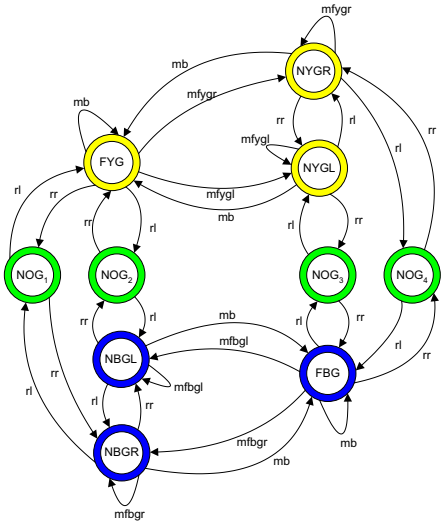

Fig. 5. Graph representation of the Topological Map

4.1 Topological Map

To construct the map, we considered 3 nodes where the BLUE goal was visible, 3 nodes where the YELLOW goal was visible and 4 nodes where no goal was visible. We used 7 different transitions to travel between these nodes.

The nodes where defined considering the relative position of the goal w.r.t the robot. We used four nodes for the situation where there is no goal on the images to avoid conflicts and ambiguities in the definition of the graph. However, the difference between these nodes can only be determined based on the history of the robot path accomplished so far.

The transitions where defined specifying the angular and/or linear speed, and the relative position which we want to maintain of a specific goal (if any) w.r.t. the robot (e.g., move forward keeping the blue goal in the right of the robot).

The graph that was used in the robots navigation is shown in Fig. 5. After defining the graph, we extracted a set of 528 training images and applied the *KL* transform. To determine how many principal components to use, we computed all

the 528 eigenvalues and used equation (2) to compute the number of eigenvalues required to achieve $\xi < 10\%$, leading to 46 principal components.

In Fig. 6, a plot comparing the mean square error for the training set and for a testing set is presented. The latter was obtained in random field positions and the error was calculated for several principal spaces with different numbers of eigenvalues, between 0 and 60.

4.2 Navigation

For localization, we used, as previously stated, the *k-nearest neighbor classifier*, with $k = 5$ an associated Euclidean metric, in the principal space. We obtained several classification results using a large number of postures from most regions of the configuration space. In Section 5 we explain some of those results.

The overall algorithm was tested by running it on the simulator, generating a new goal node randomly each time the last one was accomplished. Furthermore, we created another graph having the same nodes and transitions topology but where the images associated with each of the nodes were different — we only considered an image to be "near a goal" when it was taken at a distance of $2\,m$ from that goal, as opposing to the first situation where this distance was extended to $5\,m$ (mid-field situation). This setup also allowed us to test the percentage of times a given transition would fail for a given representation. Some of the results from those tests are presented in Section 5.

5 Experimental Results

The principal space was obtained from a set of training images. Obviously, not all images with the same size live on this space and, in general, they are far from it. In fact, by neglecting the smaller eigenvalues we are doing an acceptable approximation of the eigenspace obtained from the training images but, on the other hand, the principal space might not be so good so as to approximate conveniently other images of the same size.

In Fig. 6 we compare the mean square error for training and test sets. As expected, the error for the training set converges to a constant value. This value corresponds to the mean square distance between the test set and the eigenspace. For the training set the mean square error converges towards zero. The implemented classifier was also tested, as mentioned in Section 4, by using a large number of postures on the soccer field, with small distances between them, which ensures a good representation of the whole configuration space. We present an example of a 2 dimensional cut in the configuration space, with the coordinate y fixed in the zero value. Refer to Fig. 8 for information on the frame used.

The localization tests led to the conclusion that the classifier is not immune to the presence of outliers, which can make the path execution a bit more troublesome, justifying the need for a supervisor. However, the results are much better for 5 neighbors than the use of a single *nearest neighbor* classifier.

As for the execution of the complete navigation method, we stated that it was mostly successful in travelling between key-places of the topological map, due to

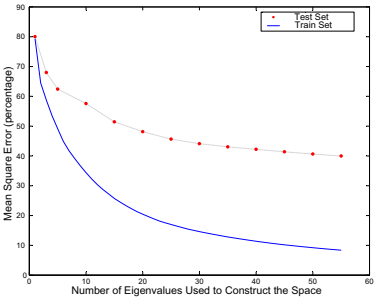

Fig. 6. Mean square error in the train and test sets

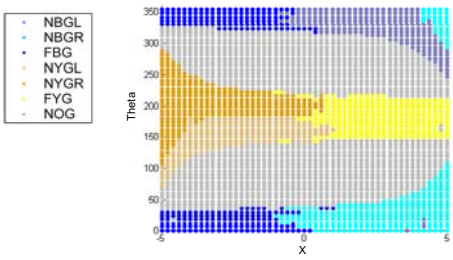

Fig. 7. Example of the classification/localization

the usage of an execution supervisor. In fact, some of the transitions proved to have a failure rate higher than 50%, which suggests that there are some implicit transitions on the graph which were not defined *a priori*. This was, in fact, expected, since the nodes did not quite correspond to the configuration space regions we intended them to represent.

A more serious situation that occurred, due to the existence of these implicit transitions, were *live-locks* — situations where the robot stays in a loop from where it cannot get out, unless a new goal node is given. Other than that, the method was always successful in achieving the proposed key-places.

In Fig. 8 we can see a trajectory of the robot in a multi-objective situation, and where the border between *Far Goal* nodes and *Near Goal* nodes was set at the distance of 2 m from the goal. It is interesting to underline angular velocity control to ensure that the robot keeps the goal on its right or left, leads the robot to align itself with the goal-posts.

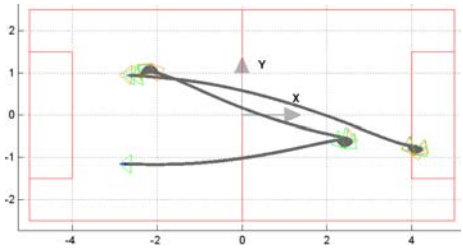

Fig. 8. Example of a trajectory

6 Conclusions and Future Work

This paper addressed the application of topological navigation methods to robotic soccer. We introduced an appearance based method which can navigate between different regions of the configuration space, represented by key-places of a topological map.

The method has shown promising results, when navigating between key-places. Although some of the transitions displayed high failure rates, the inclusion of the supervisor was able to deal with most of those situations, making the robot reach its goal nodes. However, some parts of the algorithm still need to be improved. The main one is the occurrence of *live-locks*, not in the topological map but as the result of the method application, due to specific failure cycles in the transition execution. We will investigate the application of Discrete Event Supervision techniques [9] to the detection and prevention of such cycles.

Another part of the method with open research topics is image compression. Actually, we would like to show it is possible, for this application, to use a smaller number o eigenvectors, compressing the needed information much more. This goal is most likely to be accomplished by reducing the size of the acquired images and/or using omnidirectional cameras instead of the front camera.

We are currently committed to use the developed methodology to solve the *RoboCup Challenge 4 - Play with an arbitrary FIFA ball* of RoboCup 2003 MSL. The idea is to use PCA to extract the most important features describing the ball, Topological Navigation to move the robot close to the ball and, finally, metric-based navigation already implemented in the ISocRob team to take the ball to the desired goal.

References

1. Ishiguro, H., Tsuji, S.: Image-based memory of environment. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (1996) 634–639
2. Horswill, I.: Polly: A vision-based artificial agent. Proc. Nat. Conf. Artificial Intelligence (1993) 824–829
3. Blaer, P., Allen, P.: Topological mobile robot localization using fast vision techniques. Proc. of the 2002 IEEE Int. Conference on Robotics & Automation (2002)
4. Gaspar, J., Winters, N., Santos-Victor, J.: Vision-based navigation and environmental representations with an omnidirectional camera. IEEE Trans. on Robotics and Automation **16** (2000)
5. Murakami, H., Kumar, V.: Efficient calculation of primary images from a set of images. IEEE Transactions on Pattern Analysis and Machine Intelligence **4** (1982) 511–515
6. Murase, H., Nayar, S.K.: Visual learning and recognition of 3-d objects from appearence. International Journal of Computer Vision **14** (1995) 5–24
7. Mitchell, T.: Machine Learning. 1st edn. Computer Science Series. McGraw-Hill, Singapore (1997)
8. Russel, S., Norvig, P.: Artificial Intelligence: a Modern Approach. 1st edn. Prentice Hall Series on Artificial Intelligence. Prentice-Hall, New Jersey (1995)
9. Cassandras, C., Lafortune, S.: Introduction to Discrete Event Systems. Discrete Event Dynamic Systems. Kluwer Academic Publishers (1999)