National Conference with International Participation
ENGINEERING MECHANICS 2003
Svratka, Czech Republic, May 12 – 15, 2003
paper no. 196

ROBOT NAVIGATION USING GENETIC ALGORITHM AND CASE-BASED REASONING

M. Šeda*, **J. Dvořák***

Summary: *The aim of the paper is to propose an integration of a genetic algorithm and case-based reasoning in robot motion planning in a partially known dynamic environment. The goal of the planning is to help find a path from a start to a goal position without collisions with known obstacles minimizing length and difficulty of the path. The environment is modelled by a grid in which known static obstacles or unattainable positions are defined. The robot motion is reduced in horizontal, vertical and diagonal directions. The paths realized are stored in a base of cases along with the degree of their traversability. When planning a path, first this base is searched so as to find the cases that are most similar to the given case and then they are adapted to it. If similar cases are not found in the base or adapted solutions are not good enough, a new path is searched for by a genetic algorithm.*

1. Introduction

Motion planning of a mobile robot is often decomposed into path planning and trajectory planning. Path planning (global navigation) generates a path from a start to a goal position without collisions with known obstacles. Trajectory planning (local navigation) schedules the movement of a robot along the planned path. There are many approaches to a robot navigation including genetic algorithms (GA). An advantage of GA-based approaches is their ability of adaptation to varying environment. (Nearchou 1999) and (Sugihara & Smith 1999) propose genetic algorithms for adaptive navigation in an environment represented by a two-dimensional grid map. In (Homaifar et al. 2001) a genetic algorithm for a continuous environment is described.

When an environment is dynamic and/or partially unknown, then robot navigation is a complicated task, because changes in the environment are usually very difficult to model. Therefore robots with ability to learn and explore the environment are needed. This ability is often achieved by using neural networks, reinforcement learning or evolution algorithms. *Case-based reasoning* (CBR) is another learning technique, which solves a new problem by adapting known solutions of similar previously solved problems. It seems that the CBR is a

* Doc. RNDr. Ing. Miloš Šeda, Ph.D., RNDr. Jiří Dvořák, CSc., Institute of Automation and Computer Science, Faculty of Mechanical Engineering, Brno University of Technology, Technická 2, 616 69 Brno, tel.: +420 541143332, +420 541143342, fax: +420 541142330, e-mail: seda@uai.fme.vutbr.cz, dvorak@uai.fme.vutbr.cz

suitable method for a robot control because robotic applications usually include repeated tasks.

A brief overview of CBR and its major open problems is presented in (Mántaras & Plaza 1996). (Aamodt & Plaza 1994) give an overview of the foundational issues related to CBR, and describes some of the leading methodological approaches within this field. The book (Lenz et al. 1998) in their book summarize the results of the recent years of research in CBR. In (Kruusmaa & Svensson 1998a,b) a map-based and case-based path planning are combined for a global navigation of mobile robot in an environment represented by a grid map. (Kruusmaa & Willemson 2002) analyse theoretically a possibility of covering the path space by a set of cases. Application of CBR to a global navigation in a continuous environment is described in (Haigh & Shewchuk 1994) and (Supic & Ribaric 2001). (Louis & Li 1997) combine genetic algorithms and case-based reasoning for a local navigation in a continuous environment. This problem is also solved in (Ram & Santamaría 1997) and (Chagas & Hallam 1998) by means of case-based reasoning and reinforcement learning. (Fox 2000) proposes a unified CBR architecture for robot navigation in a continuous environment, which enables both global and local navigation.

2. Path planning

Assume that path planning is considered in a rectangular plane between two locations s and g. This plane is divided into a grid of cells. For simplicity, we consider square shaped cells.

Allowed directions of robot motion are only horizontal, vertical and diagonal, see Fig. 1. In Fig. 2, the grid representation is shown. Here cells in the left upper and right lower corners represent the start and goal positions s, g and black cells correspond to obstacles. In (Sugihara & Smith 1999) hazardous obstacles are also considered. These obstacles allow a path to intersect them at the expense of a higher cost.

We consider a rectangular grid $[1, m] \times [1, n]$. A cell c of this grid is determined by a pair of coordinates in this grid: $c = (x, y)$, where $x \in \{1, 2, \dots , m\}$, $y \in \{1, 2, \dots , n\}$. A *distance* (not considering obstacles) between points $c_i = (x_i, y_i)$ and $c_j = (x_j, y_j)$ can be defined in using these formulas:

$$d(c_i, c_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{1}$$

$$d(c_i, c_j) = |x_i - x_j| + |y_i - y_j| \tag{2}$$

$$\begin{aligned} d(c_i, c_j) = \sqrt{2} \min\{|x_i - x_j|, |y_i - y_j|\} + \\ + \max\{|x_i - x_j|, |y_i - y_j|\} - \min\{|x_i - x_j|, |y_i - y_j|\} \end{aligned} \tag{3}$$

The third equation corresponds to the fact that only motions in horizontal, vertical and diagonal directions are allowed.

The robot moves on its path between adjacent cells choosing allowed directions without collisions with obstacles. That means the path is defined as a sequence of adjacent cells between s and g subject to these constraints and its total length is given by the sum of distances between adjacent cells. If there are more feasible solutions (i.e. paths between s and g satisfying defined constraints), then we try to determine the paths of a minimal value of a cost function considering both the length and the difficulty of a path.

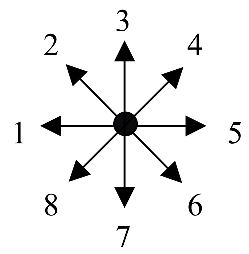

Fig. 1. Valid directions of robot motion

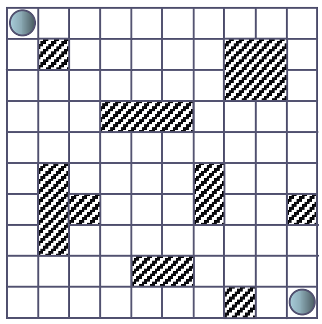

Fig. 2. Grid representation of 2D space with start and goal positions of the robot and static obstacles

3. Path planning using genetic algorithm

It is obvious that the problem is of a combinatorial nature and its time complexity depends on the granularity grid and distribution of obstacles. Even if we restrict our considerations to the case when paths have fixed lengths, the complexity remains exponential. For example, if we consider only paths with $2n$ adjacent cells, where n is the number of rows (and columns) in a square grid, then the search space contains 8^{2n} sequences of directions under consideration. Although many of them represent infeasible paths, the problem cannot be solved by enumerating all possible paths. Therefore we must use some approximation methods selecting only a part from the huge search space. In this paper we present the use of genetic algorithms. We assume that the general framework of this optimisation technique is well known and thus we will concentrate only on problem-specific settings.

For coding of chromosomes, instead of the traditional binary representation, we use a coding where each gene corresponds to direction of robot movement to the next cell. Thus each chromosome is coded by a string $S = (d_1, d_2, \ldots ,d_L)$, where $d_i \in \{1,2, \ldots ,8\}$, $i = 1,2, \ldots ,L$. As to the length of the chromosome, we choose it by the following formula

$$L = 2 * \max \{m, n\}. \quad (4)$$

In Fig. 3 is shown a path from a start to a goal position in the configuration from Fig. 2 and its coding.

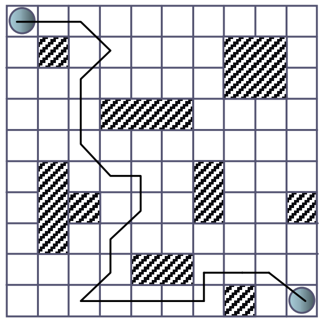

Fig. 3. A path with coding (5,5,6,8,7,7,6,5,7,8,7,8,5,5,5,5,3,5,5,6)

Coding of the path from Fig. 3 corresponds to one solution of the problem and it can be seen that this solution is not optimal. In real situations, chromosomes need not represent any solution, i.e. after the last movement represented by the last gene, the robot did not reach the goal position. Another (much more pleasant) case is when the robot reaches the goal position before passing all genes. Of course, in this case, the movements of the next genes are not taken into consideration.

When generating the sequence of directions, we must avoid all movements out of the grid and the movements that cause collisions of the robot with obstacles, e.g. the path (5,6, …) causes a collision with the obstacle in the second row and second column. As infeasible we also consider two adjacent movements that are inverse to each other, i. e. 3-7, 4-8, 5-1, 6-2, 7-3, 8-4, 1-5 and 2-6.

When executing genetic algorithm we need a tool for evaluating the quality of chromosomes. We model it by the distance of the last chromosome's gene from the goal position and a cost of this path. The two criteria are applied in the lexicographic way, i.e. that chromosome is better, whose distance from the goal position is shorter or, when distances are equal, whose cost is lower.

The described algorithm was implemented in (Sedláček 2000), where the Euclidean distance was used. Parameter settings are:

- Initial population generated randomly.
- Population size = 50.
- Binary tournament selection (each parent is determined by the most fit chromosome between randomly chosen two chromosomes from the population).
- Uniform crossover (each gene in the child solution is created by copying the corresponding gene from one or the other parent, chosen according to a binary random number generator. If the random number is 0, the gene is copied from the first parent; if it is a 1, the gene is copied from the second parent).
- Mutation of a gene selected at random.
- Incremental replacement (eliminating the chromosome with the worst fitness function)
- Termination criterion: maximal number of generations = 50000.

4. Case-based reasoning and path planning

Case-based reasoning (CBR) is based on the retrieval and adaptation of the old solutions to the new problems. A general CBR cycle may be described by the following four steps (Aamodt & Plaza 1994): (i) Retrieve the most similar case or cases; (ii) Reuse the information and knowledge in that case to solve the problem; (iii) Revise the proposed solution; (iv) Retain the parts of this experience likely to be useful for future problem solving. Besides specific knowledge represented by cases, general domain knowledge usually plays a part in this cycle by supporting the CBR processes.

In (Kruusmaa & Svensson 1998a,b) cases represent paths that the robot has traversed. A *path* $P(c_s, c_g)$ from the start cell c_s to the goal cell c_g is specified as an ordered set of adjacent grid cells

$$P(c_s, c_g) = \{c_s, \dots, c_i, \dots, c_g\}. \tag{5}$$

Each path in the casebase is stored with a value of a cost function $F(P)$, which characterises the traversability of the path:

$$F(P) = f(l, r) \tag{6}$$

The parameter l is the length of the path. The parameter r characterises the risk (or difficulty) of following the path and is calculated by gathering information during the path following process.

If, for a given start cell c_s^0 and a given goal cell c_g^0, the casebase does not contain a path leading from c_s^0 to c_g^0, a similar path is retrieved according to the formula

$$P(c'_s, c'_g) = \arg\min \left\{ F\left(P(c_s, c_g)\right) \middle| \, d(c_s^0, c_s) \le \delta, d(c_g^0, c_g) \le \delta \right\} \tag{7}$$

If the cells c'_s and c'_g are found, then a genetic algorithm is used to find paths from c_s^0 to c'_s and from c_g^0 to c'_g (this represents the adaptation step of CBR). If there are no similar cases, or if the proposed path is not good enough, the GA-based path planner has to be used to plan the whole new path. In (Kruusmaa & Svensson 1998a,b) a probabilistic searching map-based method is used instead a genetic algorithm.

After traversing the path determined in the way described above, the experience obtained is retained if necessary. It means, that this path will be stored in the casebase, only if the casebase does not contain any similar path with a lower or equal cost (similar path with higher cost will be replaced by the new one). A similarity measure for this phase is different ftrom the one used in the retrieving phase and is based on the distance between the paths P_1 and P_2 defined in this way:

$$D(P_1, P_2) = \max_{c_i \in P_1} \min_{c_i \in P_2} d(c_i, c_j) \tag{8}$$

where $d(c_i, c_j)$ denotes the distance between the points c_i and c_j. Kruusmaa & Svensson (1998a,b) use the Euclidean distance and in (Kruusmaa & Willemson 2002) for a special problem the R_∞^2-distance is used:

$$d(c_i, c_j) = \max\{|x_i - x_j|, |y_i - y_j|\} \tag{9}$$

The maxmin construction of $D(P_1, P_2)$ is generally known as *directed Hausdorff distance*. It is not the case that for any inner metrics d the directed Hausdorff distance is a real distance since it mostly fails to be symmetric; such a problem occurs, for instance, if d is the Euclidean distance. Several approaches can be taken in order to fix the problem, the most commonly used one replaces the directed distance by $\max\{D(P_1, P_2), D(P_2, P_1)\}$.

In order to keep the size of the casebase constrained it is necessary to forget the most useless cases. (Kruusmaa & Svensson 1998b) analyse the following forgetting strategies based on values of cost function:

1. Forgetting the worst cases (i.e., cases with the highest cost). The learning is thus success-driven.
2. Forgetting the average cases. The cases with the average cost are forgotten as non-characteristic to the problem at hand.
3. Forgetting the best cases. The learning is failure-driven (the robot will concentrate on not repeating its failures).

It is also possible to consider the frequency of using cases or the date of the last use and to forget cases seldom used or long unused.

The concept of representing paths and case-based reasoning can be improved in the following way. The representation of a path as a sequence of adjacent cells can be too long and therefore complicated to handle. We propose to specify a path $P(c_s, c_g)$ from the start cell c_s to the goal cell c_g as a sequence of cells $P(c_s, c_g) = \{c_s, \ldots, c_i, \ldots, c_g\}$, where the inner cells c_i of this sequence are those cells, where the direction of motion is changed or where this path intersects with another path. Therefore the path can be regarded as a sequence of line segments (c_i, c_j) and these segments are stored as cases in a graph structure, which is called a *case graph* (Haigh & Shewchuk 1994). If not only complete paths but also their parts can be reused, then the transfer rate of past experience can be considerably increased.

Let a start cell c_s^0 and a goal cell c_g^0 be given. The retrieval of similar paths can be based on the intersections of case segments with the neighbourhoods of these cells. We define a *neighbourhood* $N(c_i, \delta)$ of a cell c_i as follows:

$$N(c_i, \delta) = \{c_j \mid d(c_i, c_j) \le \delta\} \tag{10}$$

where $d(c_i, c_j)$ is the R^2_∞-distance (9). As similar paths we regard such paths, which connect points from the intersections of case segments and the neighbourhoods $N(c_s^0, \delta)$ and $N(c_g^0, \delta)$. If some similar path is found, a GA method is used to find paths from c_s^0 and c_g^0 to the corresponding case segments. For the retaining phase, we can define a similarity of case segments in this simple way: the segments (c_i, c_j) and (c_k, c_l) are similar, if it holds

$$d(c_i, c_k) \le \delta \text{ and } d(c_j, c_l) \le \delta \tag{11}$$

where d is some of the distances (1), (2), (3), (9).

5. Conclusions and future work

The paper dealt with the optimisation of the trajectory of robot moving in 2D plane. The area was represented by two-dimensional grid of cells and the motion was restricted to horizontal, vertical and diagonal direction. Some cells of the grid were occupied by obstacles and thus unattainable. With respect to the combinatorial nature of the problem studied having exponential dependence on the search space size, a solution based on a stochastic heuristic approach using genetic algorithms is proposed. We also suggest improving this path planning method by means of case-based reasoning, where segments of successfully traversed paths are reused. At present, only the genetic path planner is implemented.

In future research, the proposed case-based approach will be integrated with the GA-based path planning method. We are also going to study other representations of chromosomes and the corresponding genetic operators. It is also possible to combine GA and CBR in such a way that similar paths retrieved, that are not good enough for a direct use, will be injected into initial population in the genetic algorithm. It could increase the efficiency of path searching.

6. Acknowledgement

This paper was supported by the project MSM 262100024 "Research and Development of Mechatronic Systems" and the research project CEZ: J22/98: 261100009 "Non-Traditional Study Methods of Complex and Uncertain Systems".

7. References

Aamodt, A. & Plaza, E. (1994) Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications*, Vol. 7, No. 5, pp. 39-59.

Chagas N.C. & Hallam, J. (1998) A Learning Mobile Robot: Theory, Simulation and Practice, in: *Proceedings of the Sixth Learning European Workshop*, pp.142-154.

Fox, S.E. (2000) A Unified CBR Architecture for Robot Navigation. In Enrico Blanzieri and Luigi Portinale (Eds.) Advances in Case-Based Reasoning. 5th European Workshop, EWCBR 2000. Trento, Italy, September 6-9, Springer-Verlag, Berlin (Germany), pp. 406-417.

Haigh, K.Z. & Shewchuk, J.R. (1994) Geometric Similarity Metrics for Case-Based Reasoning. In Case-Based Reasoning: Working Notes from the AAAI-94 Workshop, Seattle, WA, August 1994, AAAI Press, pp. 182-187.

Homaifar, A., Tunstel, E., Dozier, G. & Battle, D. (2001) Genetic and Evolutionary Methods for Mobile Robot Control and Path Planning. In Ali Zillouchian and Mo Jamshidi (eds.). *Intelligent Control Systems Using Soft Computing Methodologies*, CRC Press LLC, Boca Raton.

Kruusmaa, M. & Svensson B. (1998a) Combined Map-Based and Case-Based Path Planning for Mobile Robot Navigation, in: *Proceedings of International Symposium of Intelligent Robotic Systems*, Jan 10-12, 6 pp.

Kruusmaa, M. & Svensson B. (1998b) Using Case-Based Reasoning for Mobile Robot Path Planning, in: *Proceedings of the 6th German Workshop on Case-Based Reasoning*, March 6-8, Berlin, 8 pp.

Kruusmaa, M. & Willemson, J. (2002) Covering the Path Space: A Casebase Analysis for Mobile Robot Path Planning. In: *Research and Development in Intelligent Systems XIX, Proceedings The Twenty-second SGAI International Conference on Knowledge Based Systems and Applied Artificial Intelligence*, BCS Conference Series, Springer, pp. 1-16.

Lenz, M., Bartsch-Spörl, B., Burkhard, H.-D., Wess, S. (Eds.) (1998) *Case-Based Reasoning Technology: From Foundations to Applications*. Springer-Verlag.

Louis, S.J. & Li, G. (1997) Combining Robot Control Strategies Using Genetic Algorithms with Memory, in: *Proceedings of the 6th International Conference Evolutionary Programming*, Indianapolis, IN, 13.-16. April 1997. Springer-Verlag, Berlin (Germany), pp. 431-441.

de Mántaras, R.L. & Plaza, E. (1996) Case-Based Reasoning: An Overview. *IOS Press*, Vol. 10, No. 1 pp. 21-29.

Nearchou, A.C. (1999) Adaptive Navigation of Autonomous Vehicles Using Evolutionary Algorithms. *Artificial Intelligence in Engineering*, Vol. 13, No. 2, pp. 159-173.

Ram, A. & Santamaría, J.C. (1997) Continuous Case-Based Reasoning. *Artificial Intelligence*, Vol. 90, No. 1-2, February, pp. 25-77.

Sedláček, M. (2000) Optimalizace trajektorie robota. *Diplomní práce.* VUT FSI, Brno, 61 str.

Sugihara, K. & Smith, J. (1999) Genetic Algorithms for Adaptive Planning of Path and Trajectory of a Mobile Robot in 2D Terrains. *IEICE Transactions on Information and Systems*, Vol. E82-D, No. 1, pp. 309-317.

Supic, H. & Ribaric, S. (2001) Adaptation by Applying Behavior Routines and Motion Strategies in Autonomous Navigation. In David W. Aha and Ian Watson (Eds.). *Case-Based Reasoning. Research and Development. 4th international Conference on Case-Based Reasoning, ICCBR 2001*, Vancouver, BC, Canada, July 30 - August 2. Springer-Verlag, Berlin (Germany), pp. 517-530.