ELECTRONICS AND ELECTRICAL ENGINEERING
ISSN 1392 – 1215 2007. No. 6(78)
ELEKTRONIKA IR ELEKTROTECHNIKA

AUTOMATION, ROBOTICS
T125
AUTOMATIZAVIMAS, ROBOTECHNIKA

Learning Possibilities in CPN

V. Baranauskas, K. Šarkauskas
Department of Control Engineering, Kaunas University of Technology,
Studentų st. 48-319, LT-51367 Kaunas, Lithuania; phone: +370 37 300292; e-mail: virgisbar@yahoo.com,
Studentų st. 48-107, LT-51367 Kaunas, Lithuania; phone: +370 61025585, e-mail: Kastytis.Sarkauskas@ktu.lt
S. Bartkevičius
Department of Theoretical Electric Engineering, Kaunas University of Technology,
Studentų st. 48, LT-51367 Kaunas, Lithuania; phone: +370 37 300253; e-mail: Stanislovas.Bartkevicius@ktu.lt

Introduction

Term "learning" means that the system, independent from any human actions, learns itself and improves it's knowledge. Term "learning" is different from the term "training". The main difference is that the term "training" means the dependence from human, who controls system or process operations. When some conflict emerges, human decides what solution should be admitted for safe further system work. The decision of current situation is written to the memory. If the same situation emerges in further system work, the right solution is already known and it will be used. In the learning system, human only supervises conflict situations and how specified software solves emerged problems. There, software is in influence from any human actions.

Mobile robots are very popular now. Learning processes are very important in robotics. Scientists have been fascinated by mobile robots for many years; machines have been designed with wheels and tracks or other locomotion devices and/or limbs to propel the unit. When the environment is well ordered these machines can function well and have demonstrated their ability to carry out useful work [1].

Numerous wheeled machines (for manufacturing applications) have also been commercialized successively automatically to transport components and products from one site to another using various navigation methods, such as simply following lines on the floor or using more sophisticated systems based on beacons fixed around the working site; the automobile industry is a good example of such machines [1].

Mobile robots are used in industry (walking forest machines, rescue robots), military (mobile detection assessment response system (MDARS) exterior robot, spiral track autonomous robot (STAR)) and house cleaning (IFF facade cleaning robot) [1, 2, 3]. There are mobile robots, which has degrees-of-freedom (DOF) and could walk [3].

There is one big problem in mobile robots navigation – obstacle avoidance. This paper describes how mobile robots can navigate and learn to navigate in different environments. Authors of the article, suggests their solution for mobile robots collision avoidance with a static objects.

Obstacle avoidance and learning

Mobile robots must navigate among stock-still or moving obstacles without collisions with them. Obstacle avoidance in mobile robots and/or manipulators is a large and extensively researched topic. Many techniques have evolved attempting to achieve this task efficiently [4]. The mere way to avoid collisions with stock-still objects is the usage of GPS. This system is useful in open space, but if robot is in indoors or underground environments, GPS cannot operate there [5].

One such technique of obstacle avoidance is artificial potential fields. The main importance is represented by a fuzzy variable, which in turn is derived form robot parameters: separation distance, separation angle and robot speed. These parameters are converted into variables by dividing the area, surrounding the robot, into fuzzy sectors (Fig.1). Each sector is assigned to a value defining its importance [4, 6].

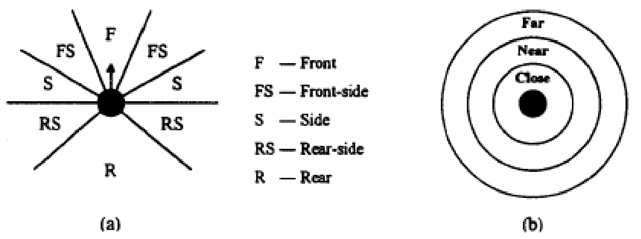

Fig. 1. The area, around the robot, is divided into sectors, (a), or distances, (b), which are then rated in terms of importance

Luciano Pimenta and his colleagues, suggests using electrostatic field computation for robot navigation. The navigation function computation uses finite elements methodology. They consider that the target boundary conditions are constant Dirichlet with value equals to zero, and obstacles boundary conditions are also constant Dirichlet with identical positive values. This implies that the negative gradient of the resultant field close to obstacles is perpendicular outward of the obstacles surface [7]. The biggest problem of this computation is huge mathematical calculations of electrostatic fields.

If the surrounding is unknown, mobile robot could have sensors. Sensors collect information about the environment and adapt robot motions to any new contingency or event. The vehicle executes the motion and the process restarts. Such system has modelling module, planning module, reactive navigation module and architecture of integration [8].

If the surrounding and environments is known, it is possible to use other methods. For example, the beam curvature method. It generates motion commands in two separate steps. In the first step a desired local goal heading is determined by using a directional approach that is called beam method. In the second step, the steering commands yield a motion in the desired local direction by using CVM method and taking into account the dynamic constraints of the robot. The local trajectory toward the objective is calculated in real time for each sampling interval [9].

Andrew Seely is the one, who suggests to use Petri nets with neural networks [10]. It is Petri net threshold learning unit. There no modification to the Petri net structures is required; all threshold-learning units processing is accomplished through manipulation of the Petri net arc connectivity between places and transitions that are created to serve as thresholds and weights. The PNTLU addresses a problem domain that is purely in the realm of neural networks while conforming strictly to a basic Petri Net definition, and so may be more appropriate than technologies like the FPN or FNPN in artificial intelligence applications where Petri Net structures are desirable.

Learning in coloured Petri nets

Authors of the article suggest creating learning system in coloured Petri nets [11]. There are some requirements for such system creation in coloured Petri nets. They are: requirement for data capture and requirement for this data accessibility in the entire project.

The declarations of model variables are used in coloured Petri nets. It could be used for database creation. Specialized software CENTAURUS has been modified for purposes of data accessibility. Global variables are used in it [12, 13] and the model ffunctionality in 2D is realized. So, software CENTAURUS could be used for the development of learning systems.

Why does learning problem appear? Let's analyse example of manufacture line with attendant robots. Robots transport details and products. There emerges problem, how to avoid robots collisions with static and dynamic obstacles. The amount of the traces could be big, according to the amount of robots.

It is possible to organise system training for such situation. During system training, emerges new problem. It is big amount of the traces and each of them must be checked. So manual training is very monotonously and difficult procedure. The rules and procedures, formed for manual training, compose all possibilities for system learning.

Training procedure in known surrounding could be realised in two ways: to form all mediate points for known movement trajectory or change the movement trace during robot training, forming dynamic database of the trace. Special function is realized for this purpose

fun FixTrace(trc:way, regnam: string): bool.

Its functionality depends from the changes of movement trace. When the trace is formed, it is written to the declarations (database), using special function fun FixDecl, and could be used in robots training and learning.

This data exchange system, composed for system training, do not require any programmable changes applying it for learning systems.

The movement of the robots is organized in two ways: using contact sensors (different touch sensors) and using scanning systems (optical, hypersonic, laser sensors). When robot moves to the goal, it is not must or could be in the visible zone. If robot wants to reach the goal in the rational way, it must change the movement trajectory. In this learning level, the main task of the robot is to identify the strategy for obstacles avoidance. Learned trace must be rational in movement distance or leeway time.

It is proved, that for the rational formulation of the trace, more amount of information must be used. Authors of the article made assumption: when robot moves in known surrounding, it is effectively to form the trace, using surrounding map, search could be done using scanners.

The surrounding map identifies more rational ways. Especially, when there are few possible ways to reach the goal. If there is no surrounding map, learning must be done like in unknown surrounding by the test and mistakes methodology.

The search, using scanners allows us to form more rational movement trajectories till the obstacle or moving around it. We could say that, using this method, it is possible to solve some problems of forming the trace. One of them is, that the trace must be formed optimally.

Harmonise of these two methods allows to solve tasks, that couldn't be solved without test and mistakes.

Strategy of the suggesting learning method is shown in Fig. 2 and it is:

1. The whole surrounding is divided to the quadrates, not considering to the objects that are inside of them. It simplifies identification problems of the map (known surrounding). This dividation is used for all cases of trajectories.
2. Quadrate, where the goal is, has zero identification. Quadrates, which adjoin to it, has first identification, successive identifications are one unit bigger, than the last and so on. The numerical meaning of the quadrate could be called as we wish: weight coefficient, gradient coefficient. It doesn't change the meaning, because direction of the goal is form quadrate to quadrate following the decrease of coefficients.
3. Robot scans space. If the goal is not seen through the obstacle, scanner identifies breakpoints. Breakpoints corresponds various openings in obstacles, or its edges.

4. If several breakpoints exist, robot goes to the quadrate, with the smallest identification coefficient. If identification coefficients are equal, robot goes to quadrate, which is closer.
5. The 3 and 4 subsections are repeated until the goal is reached.
6. When the goal is reached, function FixTrace writes all mediate points coordinates to learning database.

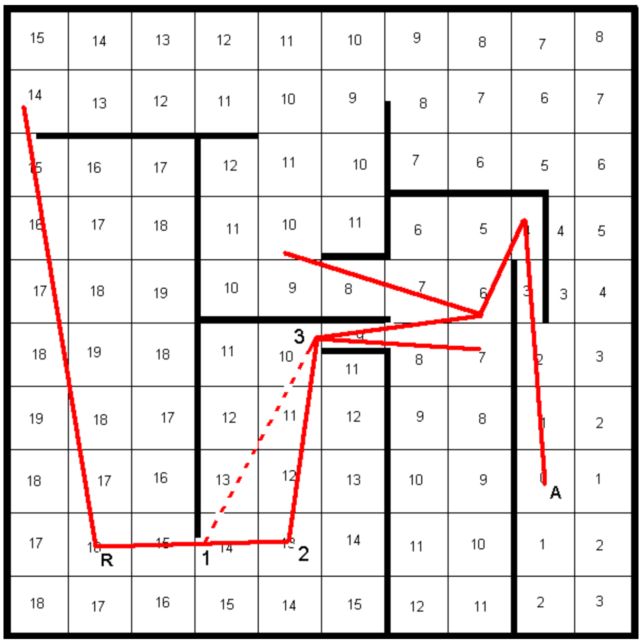

Fig. 2. Strategy of the suggested learning method

Figure 2 shows trace search according to described algorithm. The goal (point A) generates weigh identifications of the quadrates. Successive adjoined identifications are one unit bigger, than the last. Weigh coefficients are not transferred through walls. Robot (point R) scans surrounding and identifies two breakpoints for the shown situation. In the direction of breakpoints, robot searches quadrate, with the smallest weigh coefficient. If emerges situation, that there are quadrates, with the same weigh coefficients, in two directions, robot goes to the quadrate, which is closer. Robot goes to the quadrate, with the weigh coefficient 13 in the figure 2. There the scanning process continues in all possible directions, except the direction, in which robot moved to the quadrate. The procedure of scanning and moving is repeated, until the quadrate with the zero identification is reached.

It is important to notice, that the trajectory of the learned trace isn't optimal and optimization of it could be done, while changing trajectory of the robot movement. It isn't the rational way for problem solving and authors of the article suggest modified method for it. This method is shown in figure 3. After the scanning, in the direction of the breakpoint, software fixes points, depending on the clearances of robot and its cargo. New points are fixed for the further selection. Firstly considering to the weight coefficients, and if it is equal, then to the distance.

The doted line in Fig. 2 shows search of the trace, according to optimal algorithm.

Robot moves in topologic space, defined by X_i sets Dekart product. It is enough to identify two coordinates for robot movement. The first set is Euclidean set coordinate X:

$$\mathbf{X} = \{(x) : x \in \mathbf{X}\}. \tag{1}$$

The second set is Euclidean set coordinate Y:

$$\mathbf{Y} = \{(y) : y \in \mathbf{Y}\}. \tag{2}$$

Robot being place, at any time, is described with sets of X and Y in Dekart product in Euclidean space:

$$\mathbb{R}^2 = (\mathbf{X} \times \mathbf{Y}) = \{(x,y) : x \in \mathbf{X}, y \in \mathbf{Y}\}. \tag{3}$$

Robot moves in straight sections, so its way, between two points, is defined in Euclidean distance.

$$\begin{cases} \left\| (x_2, y_2) - (x_1, y_1) \right\|, \\ E_d((x_1, y_1), (x_2, y_2)). \end{cases} \tag{4}$$

If the coordinates of obstacle is (x,y), during optimization, new Euclidean space coordinates, assessing robot clearance, is formed

$$(x_1, y_1) = (x + R_x, y + R_y), \tag{5}$$

where Rx, R_y are robot clearance coordinates.

The optimal condition is defined from Euclidean distances. This distance must be the shortest

$$E_d((x_1, y_1), (x_3, y_3)) < E_d((x_1, y_1), (x_2, y_2)) + \\ + E_d((x_2, y_2), (x_3, y_3)). \tag{6}$$

Robot (point R) scans surrounding and sets two breakpoints to the current situation. According to the breakpoints direction, points, evaluating clearance of the robot, are fixed. In the first step, the quadrates, with the same identifying coefficient, are found. Then the robot goes to the quadrate, which is closer to it. There the scanning process continues in all possible directions, except the direction, in which robot moved to the quadrate. The procedure of scanning and moving is repeated, until the quadrate with the zero identification is reached. This variant of trace formulation does not require any optimization of the trace.

Conclusions

1. The special software, based on coloured Petri nets in 2D graphic environment was created. This software allows us to identify moving objects conflicts with static objects in known surrounding and composes optimal trajectories of movement.

2. Learning process is realized in the special software. Learning data is captured and stored in database. This database could be used by any moving object, trying to move on trajectory, which was learned previously.

3. All software assumptions, for mobile robots learning in automatic mode, while changing beginning and end points, are composed.

References

1. **Virk G. S.** Industrial mobile robots: the future // Industrial Robot: An International Journal. – 1997. – Volume 24, No. 2. – P. 102–105.
2. **Pransky J.** Mobile robots: big benefits for US military // Industrial Robot: An International Journal. – 1997. – Volume 24, No. 2. – P. 126–130.
3. **Rooks B.** Mobile robots alk into the future // Industrial Robot: An International Journal. – 2002. –Volume 29, No. 6. – P. 517–523.
4. **McFetridge L., Yousef Ibrahim M.** New technique of mobile robot navigation using a Hybrid Adaptive Fuzzy-potential field approach // Computers ind. Eng. – 1998. – Volume 35, No. 3–4. – P. 471–474.
5. **Ashkenazi V., Park D., Dumville M.** Robot positioning and the global navigation satellite system // Industrial Robot: An International Journal. – 2000. – Volume 27, No. 6. – P. 419–426.
6. **Cosio A. F., Castaneda P. A.** Autonomous robot navigation using adaptive potential fields // Mathematical and Computer Modelling. – 2004. – Volume 40. – P. 1141–1156.
7. **Pimenta L. C. A., Fonseca A. R., Pereira G. A. S., Mesquita R. C., Silva E. J., Caminhas W. M., Campos M. F.M.** Robot navigation based on electrostatic field computation // IEEE Transactions on magnetins. – 2006. – Volume 42, No. 4. – P. 1459–1462.
8. **Minguez J., Montano L.** Sensor-based robot motion generation in unknown, dynamic and troublesome scenarios // Robotics and Autonomous Systems. – 2005. – Volume 52. – P. 290–311.
9. **Fernandez J. L., Sanz R., Benayas J. A., Dieguez A. R.** Improving collision avoidance for mobile robots in partially known environments: the beam curvature method // Robotics and Autonomous Systems. – 2004. – Volume 46. – P. 205–219.
10. **Seely A.** The Petri Net Threshold Learning Unit // Nova Southeastern University. – 2002. – P. 1–11.
11. **Bartkevičius S. K., Mačerauskas V., Šarkauskas K.** Spalvotųjų Petri tinklų taikymas valdymo sistemoms modeliuoti // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2003. – Nr. 4(16). – P. 7–11.
12. **Baranauskas V., Bartkevičius S., Kragnys R., Šarkauskas K.** Modeling Control Systems with Coloured Petri Nets Using Global Variables // Electrical and control technologies 2006: Proceedings of international conference. – Kaunas: Technologija, 2006. – P. 306–309.
13. **Bartkevičius S., Kragnys R., Šarkauskas K.** Global Variables in Colored Petri Nets // Elektronika ir elektrotechnika: Mokslo darbai. – Kaunas: Technologija, 2006. – No. 5(69), – P. 49–52.

Submitted for publication 2007 03 01

V. Baranauskas, K. Šarkauskas, S. Bartkevičius,. Learning Possibilities in CPN // Electronics and Electrical Engineering – Kaunas: Technologija, 2007. – No. 6(78). – P. 71–74.

In the flexible lines, when it is frequently necessary to change the sequence of technological process, the use of mobile robots is preferred. Training of the mobile robots in known surrounding is required for the efective functionality of control systems. Training proces is very long, so a suggestion emerges to use robots learning in known surrounding for this problem. For purposes of training and learning, the colored Petri nets are suitable. There are used global variables, it is possible to form, to accumulate and to preserve data. Authors created a special system for mobile robot learning. It uses scanner and known parameters of the surrounding. Learning of the system is implemented in the model, the search and the conflict situations are solved in the graphic subsystem of the software. Learning data is captured to the database, which is common to all mobile robots; it is in the declarations of the colored Petri nets. Optimization of the traces solves during the search or the trace. It allows us to achieve good learning results quickly. Ill. 2, bibl. 13 (in English; summaries in English, Russian and Lithuanian).

В. Баранаускас, К. Шаркаускас, С. Барткевичюс. Возможности самообучения в цветных Петри сетях // Электроника и электротехника. – Каунас: Технология, 2007. – № 6(78). – С. 71–74.

В гибких линиях, когда часто надо менять последовательность технологического процесса предпочтительно использование мобильных роботов. Одним из путей повышения эффективности функционирования является начальное обучение мобильных роботов. Для обучения тратится много времени, так что желательно использование самообучения в известной окружающей обстановке. Для целей обучения и самообучения хорошо подходят цветные Петри сети, в которых, используя глобальные переменные, и возможно формировать, накапливать и сохранять данные самообучения. Самообучающаяся система использует данные известной окружающей среды и как основной элемент поиска использует сканирующие системы. Самообучение проводится на модели, поиск и конфликтные ситуации решаются на графической подсистеме программного пакета, а результаты самообучения накапливаются в общей базе данных, т. е. в декларациях цветных Петри сетей. Вопросы оптимизации трас решаются во время поиска трасы и это, позволяет достичь достаточно хороших результатов самообучения. Ил. 2, библ.13 (на английском языке; рефераты на английском, русском и литовском яз.).

V. Baranauskas, K. Šarkauskas, S. Bartkevičius. Apsimokymo galimybės spalvotuosiuose Petri tinkluose // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2007. – Nr. 6(78). – P. 71–74.

Lanksčiosiose linijose, kuriose kartkarčiais reikia keisti technologinio proceso eigą, naudojami mobilieji robotai. Kad valdymo sistemos efektyviai funkcionuotų, mobiliuosius robotus reikia apmokyti žinomoje aplinkoje. Apmokymui sugaištama daug laiko, todėl natūraliai kyla pasiūlymas dėl mobiliųjų robotų apsimokymo žinomoje aplinkoje. Valdymo sistemoms apmokyti ir apsimokyti labai gerai tinka spalvotieji Petri tinklai, kuriuose galima formuoti, kaupti ir išsaugoti apsimokymo duomenis, naudojant globalius kintamuosius. Mobiliųjų robotų apsimokymui sukurta sistema, besinaudojanti žinomos aplinkos parametrais ir kaip pagrindinį paieškos elementą naudojanti skenavimo sistemą. Sistema apmokoma modelyje, konfliktines ir paieškos situacijas sprendžiant grafinėje programų paketo posistemyje, o apmokymo rezultatai kaupiami bendroje visiems mobiliesiems robotams duomenų bazėje, t. y. spalvotųjų Petri tinklų deklaracijose. Trasų optimizavimo klausimai sprendžiami trasos paieškos metu ir tai leidžia gana greitai pasiekti pakankamai gerus apsimokymo rezultatus. Il. 2, bibl. 13 (anglų kalba; santraukos anglų, rusų ir lietuvių k.).