International Journal of Fuzzy Logic and Intelligent Systems, vo . 7, no. 2, June 2007 pp. 138-142

Path planning of a Robot Manipulator using Retrieval RRT Strategy

Kyongsae Oh[1], Euntai Kim[1] and Young-Wan Cho[2]

1 Department of Electrical and Electronic Engineering, Yonsei University ,
134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea
etkim@yonsei.ac.kr
2 Department of Computer Engineering, Seokyeong University,
16-1 Jungneung-dong, Sungbuk-ku, Seoul 136-704, Korea

Abstract

This paper presents an algorithm which extends the rapidly-exploring random tree (RRT) framework to deal with change of the task environments. This algorithm called the Retrieval RRT Strategy (RRS) combines a support vector machine (SVM) and RRT and plans the robot motion in the presence of the change of the surrounding environment. This algorithm consists of two levels. At the first level, the SVM is built and selects a proper path from the bank of RRTs for a given environment. At the second level, a real path is planned by the RRT planners for the given environment. The suggested method is applied to the control of KUKA™, a commercial 6 DOF robot manipulator, and its feasibility and efficiency are demonstrated via the cosimulatation of MatLab™ and RecurDyn™.

Key Words : Path planning, RRT, 6 DOF manipulator, SVM

1. Introduction

Over the last decades, the general manipulation planning problem received much attention within the robot community. Especially, path planning for robot manipulators which move objects among obstacles is certainly among them. In the manipulation path planning, the manipulator tries to plan its motion such that bodies of the manipulator or the whole platforms do not collide with each other or the given obstacles. In this paper, we deal with the problem of manipulation control where the manipulator moves a target from a start configuration to a goal configuration without collision under the changes of the task environment.

The algorithm consists of two levels. At the first level, a support vector machine (SVM) structure [1,2,3,4,5] is employed to select a proper path from the bank of RRTs which have been already constructed. Or, if the selection fails, it commands the second level to generate a new RRT for a given task environment. At the second level, a path is planned by a rapidly-exploring random tree (RRT) planners [6,7,8] such that the manipulator does not collide with obstacles or the robot itself. The RRT is a quick and efficient search algorithm for high dimension constrained problem and has been used in many fields such as robotics, assembly analysis, virtual prototyping, pharmaceutical drug design, manufacturing, and computer animation.

In the suggested method two levels are combined and the algorithm is named as the RRS. When the environment is new, the RRS generates a new RRT. But if not, the SVM level retrieves a path from the bank of RRTs instead of generating a new one. The binary hierarchical tree-one against all (BHT-OAA) strategy [5] is used in the SVM level to select the proper RRT.

The rest of the paper is organized as follows: A brief review of the RRT and the SVM is given in Section 2. In Section 3, the SVM level based on the BHT-OAA strategy is presented and the RRS algorithm is proposed. In Section 4, the comsimulation environment composed of MatLab™ and RecurDyn™ is set up and the suggested method is applied to the control Kuka robot, a commercial 6 DOF robot manipulator. In Section 5, some conclusions are drawn.

2. Preliminaries: Rapidly-exploring random tree and support vector machine

2.1 RRT algorithm

The RRT is a popular path planning algorithm and has been used in many applications such as robotics, assembly analysis, virtual prototyping, pharmaceutical drug design, manufacturing, and computer animation. The main idea is that in the state space, sampling points are made toward unexplored region and then the tree is constructed toward a sampling point that is a new vertex without collisions. Therefore the RRT finds a proper path for robot manipulator. This algorithm constructs a path through several steps. In each step, a simple iteration is performed to

Manuscript received Feb. 21, 2007; revised Jun. 12, 2007.
This research (paper) was supported by SUNTRONIX, INC.

extend the tree by adding a new vertex toward a randomly selected state.

Figure 1 shows the basic RRT construction algorithm. In the first step, the sample vertices are constructed randomly around the tree. In the second step, the nearest vertex is chosen and then the tree is extended to the vertex. The extension can have three possible results. The first one is "Reached" and it means that the tree reaches the vertex successfully. The second one is "Advanced" and it means that the tree fails in reaching the chosen vertex and the tree needs more vertices to reach the nearest vertex. The last one is "Trapped" and it means that the tree cannot reach it within the margin or the tree makes a collision. For many problems, the RRT can be performed quickly using incremental distance computation algorithms.

```
BUILD_RRT(x_init)
T ← (x_init);
for k = 1 to K do
    x_rand ← RANDOM_CONFIG();
    EXTEND(T, x_rand);
Return T

EXTEND(T, x)
x_near ← NEAREST_NEIGHBOR(T, x);
if NEW_CONFIG(x, x_near, x_new) then
    add_vertex(T, x_new);
    add_edge(T, x_near, x_new);
    if x_new = x then
        Return Reached;
    else
        Return Advanced;
Return Trapped;
```

Fig. 1. The basic RRT construction algorithm [8]

2.2 SVM

The SVM is a popular supervised learning algorithm for pattern classification and nonlinear regression. The basic idea of SVM is to maximize the margin of the separating hyperplane in the input space. Training samples have the binary labels, +1 or -1. The SVM finds the optimal hyperplane which separate the two groups of samples by maximizing the margin of the hyperplane. The QP optimization is employed to find the optimal parameters of the hyperplane.

Originally, the optimal hyperplane is a linear classifier. But, for nonlinear classification, the nonlinear kernel trick can be used as in [5].Therefore the maximum margin hyperplane can be applied to the feature space whose dimension is higher than that of the input space. Several nonlinear functions can be used as a kernel function and a radial basis function (RBF) is certainly among them. In the RBF, the activation decreases monotonically with a distance from a central point as shown in Figure 2.

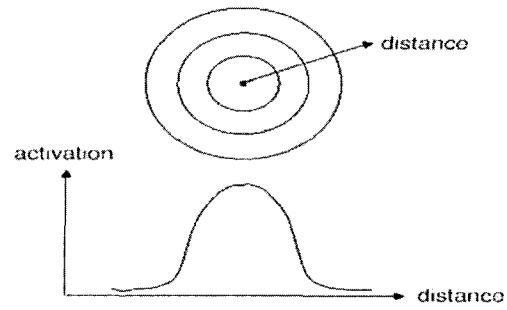

Fig. 2. Radial Basis Function

3. Retrieval RRT Strategy

3.1 Binary Hierarchical Tree – One Against All

The RRS is composed of two levels. At the first SVM level, we employ the binary hierarchical tree-one against all (BHT-OAA) strategy [5] to implement the multiclass classification. Figure 3 represents the architecture of the BHT-OAA strategy.ture is represented by fig 2.

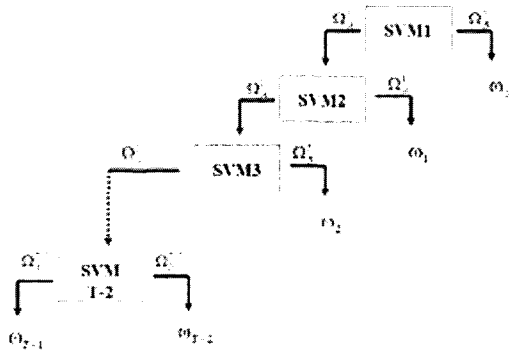

Fig. 3. Examples of BHT-OAA for a T-class classification problem

In the figure, each node in the box corresponds to the SVM and discriminates between two groups of classes and , where represents the information class. Namely, has the information to find each SVM solves a binary class problem defined by one information class (e.g., $w_i \in \Omega$) against all the others, i.e.,

$$\begin{aligned} \Omega_A &= w_i \\ \Omega_B &= \Omega - w_i \end{aligned} \tag{4}$$

The BHT-OAA strategy is summarized as follows:

```
-----------------------------------------
Step 0 : Root Node
- Set level index = 0
- Divide Ω into two group Ω and Ω
such that
Step 1 : Level Branching
- Divide Ω into two group Ω and Ω
such that
- Set level index = +1
Step 2 : Stop Condition
- If Card( Ω ) ≥ 2, go to Step 1. Otherwise Stop.
-----------------------------------------
```

3.2 Algorithm

In this subsection, the RRS is proposed for the robot motion planning by combining the SVM and the RRT. The proposed RRS extends the RRT framework to deal with the change of the task environment.

The suggested method is composed of two levels. At the first level, BHT-OAA equipped with the SVM is employed to select a proper RRT. In this process, SVM is already trained via some path. In other words, SVM is trained by the RRT bank. At the second level, the selected RRT is applied to the control of a robot or a new RRT is generated. Initially, an RRT is generated for a given environment and saved in the RRT bank. If another task is given and its environment is similar to one of the stored environments, the SVM level selects a proper RRT according to the BHT-OAA rule and the proper RRT is retrieved from the bank of RRTs. The role of the SVM level is to build the separating nonlinear hyperplane in the space of environment space, thereby classifying the environment parameters for each RRT. Namely, SVM finds the proper path in the RRT bank. If SVM doesn't find it, RRT construct new path and then SVM is trained by new path.

As more environments are stored in the RRT banks, the suggested method accomplishes the given task without generating the new RRT. The pseudo code of the RRS is given as follows.

```
-------------------------
BUILD_BHT-OAA(DB)
1. Divide two group within DB using SVM;
2. Store parameter of SVM;
-------------------------
-------------------------
FIND_PATH
2. BHT-OAA(Input)
3. If Collision then
4    Path = RRT(Input); // Construct new path
5.   BuildDB (Path ,Input)// Store new path in DB
6.   BUILD_BHT-OAA(DB );
7.   Goto STEP1
8. else
9.   BuildDB (Input)// Use the path in DB
10.  BUILD_BHT-OAA(DB );
11.  Return SUCCESS;
-------------------------
```

Fig. 4. The RRS Algorithm

4. Cosimulation and consideration

In this section, we apply the suggested scheme to the control of a Kuka manipulator. The Kuka manipulator is a popular 6 DOF commercial robot developed by the KUKA Robot Group [10].

4.1 Cosimulation and considerartion

In this simulation, we set up the cosimulation environment to implement the suggested the RRS in the control of the Kuka. The environment consists of MatLab™ and Simulink made by MathWork™ and RecurDyn™ made by FunctionBay™ [11]. MatLab™ and Simulink™ provide the cosimulation framework for the control of Kuka robot and RecurDyn™ is used to model the kinematic model of the Kuka robot.

For cosimulation, the RecurDyn™ model of the Kuka robot is embedded in the closed loop of the Simulink as shown in Fig. 4 and the parameters of the Kuka model in RecurDyn™ are used in the MatLab Robotics Toolbox[13], thereby guaranteeing that both RecurDyn™ and MatLab™ models produce the same robot motion. In the cosimulation,,the Motion Strategy Library (MSL) in [14] is modified and used in the implementation of the RRT. The cosimulation is carried out on a 2.8GHz Pentium IV, running Linux and Window XP

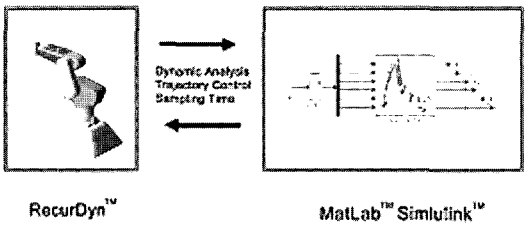

Fig. 5. Block diagram of combining with RecurDyn™ and MatLab™

In the cosimulation,,the Motion Strategy Library (MSL) in [14] is modified and used in the implementation of the RRT. The cosimulation is carried out on a 2.8GHz Pentium IV, running Linux and Window XP.

4.2 Considerartion of RRS

The scenario considered in this paper is that a robot manipulator moves four boxes on the table to a nearby shelf. For simplicity, we use the size of boxes the robot move as the environment parameters in this example. Two paths for the small and big boxes are constructed by the RRT and stored in the RRT bank. Then, the robot manipulator tries to move a medium box on the table to the shelf. The RRS retrieves a proper path from the bank of RRTs, accomplishes the task and renewes the BHT-OAA. The robot tries to move a new box again but the fourth box is very big. The RRS fails in accomplishing the give task and generate a new path by RRT.

Figure 5 shows our simulation results displayed by RecurDyn™. (a) is a small box and (b) is a big box. Paths of (a) and (b) are assumed to be strored in the bank of RRTs.

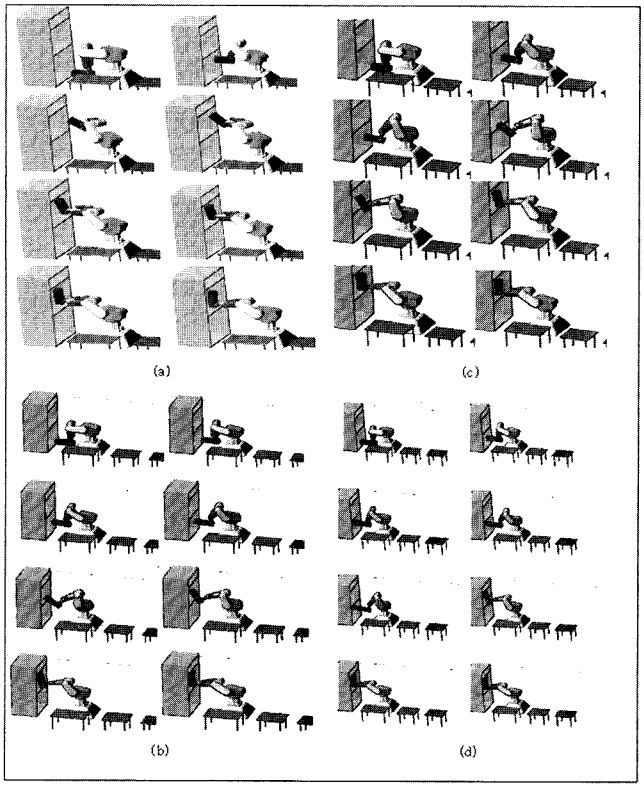

Fig. 5. RRS simulation:
(a) small box (b) big box in the bank of RRTS and (c), (d) is new task

(c) and (d) are new situations and their paths are not stored in the bank. Initially, the RRS searches for a proper path for each case. In case (c), the SVM level in decides to use the path of case (b) in the bank and it works. In case (d), however, the RRS fails in finding a path and decide to command the second level to generate a new path by RRT. As the bank of the RRT gets larger, the suggested RRS fulfills the given new task without generating a new path. The SVM level separates the environment parameters by using the nonlinear kernel.

Table 1 shows the result that a time of RRS found a proper path in memory and of RRT find new path in that case of medium box. The decreasing time rate was 83.3%.

Table 1. Comparison of RRT and CMACRRT

Method	Time (sec.)
RRT	0.281
RRS	0.047

5. Conclusion

We have successfully designed and implemented an algorithm which plans the robot motion in the presence of the change of the surrounding environments. This was done by combining the RRT framework with the multiclass SVM based on the BHT-OAA strategy. When the environment is new, the RRS generates a new RRT. But if not, the SVM level retrieves a path from the bank of RRTs instead of generating a new one. The binary hierarchical tree-one against all (BHT-OAA) strategy is used in the SVM level to select the proper RRT.

The suggested method was applied to the control of Kuka robot and its validity was shown in the cosimulation of the RecurDyn™ and Simulink.

Bayesian Network has problems like concentration of activity node, too many sensor and sensor relationship nodes, loss of activity that actually happened, and too short quantization time interval in activity recognition. So, in this paper, we propose the new activity recognition methods that has actually happened activity nodes, actually activated sensor nodes, BN structure derived from structure learning K2 algorithm, and reasonable quantization time interval. And from this proposed activity recognition method, BN has less complexity and provides better activity recognition rate.

References

[1] S.R. Gunn, *Support Vector Machines for Classification and Regression*, ISIS Technical Report, University of Southampton ,May 1998.

[2] V. Cherkassky, "The Nature of Statistical Learning Theory," *IEEE Trans. on Neural Networks*, vol. 8, no. 6, pp. 1564-1564, Nov. 1997.

[3] V. Vapnik, *The Nature of Statistical Learning Theory* Springer, New York, 1995.

[4] V. Vapnik, V. "An Overview of Statistical Learning Theory," *IEEE Trans. on Neural Networks*, vol. 10, no. 5, pp. 988-999, Sept. 1999.

[5] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Trans. on Geoscience and Remote Sensing*, vol.42, pp. 1778-1790, Aug. 2004.

[6] S. M. LaValle and M. Steven, " *Planning Algorithms*," Cambridge University Press, 2006.

[7] B. R. Donald, K. M. Lynch and D. Rus, *Algorithmic and Computational Robotics: New Directions*, *Wellesley*, 2001.

[8] S. M. LaValle and J. J. Kuffner, "RRT-connect: An efficient approach to single-query path planning," *In: Proc. of IEEE International Conf. on Robotics and Automation*, pp. 995-1001, 2000.

[9] L. Bottou, C. Cortes, J. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeCun, U. Muller, E. Sackinger, P. Simard and V. Vapnik, "Comparison of classifier methods: A case study in handwriting digit recognition," *In: Proc. of International Conf. on Pattern Recognition*, pp. 77-87, 1994.

[10] KUKA manipuator, http://www.kuka.com/en/

[11] FuntionBay, Inc., http://www.functionbay.co.kr/

[12] S. Park, K. Oh and E. Kim, "Cosimulation of thecontrol 6-DOF Kuka manipulator by simulink and Recurdyn," *International Technical Conf. on Circuits/Systems, Computers and Communications*, vol. 1, pp. 271-272, July 2005.

[13] Robotics Toolbox for MATLAB (Release 7), http://www.cat.csiro.au/ict/staff/pic/robot/

[14] The Motion Strategy Library(MSL) at University of Illinois, http://msl.cs.uiuc.edu/~lavalle/

Kyongsae Oh was born in Seoul, Korea, in 1979. He received in B.S. and M.S. degree in electronic engineering from Yonsei University, Seoul, Korea, in 2005, and 2007, respectively. He is an assistant engineer of image development team of system LSI division in semiconductor business of Samsung electronics. His current research interests include biometric recognition and digital image process.

Youngwan Cho was born in Hamyang, Korea, in 1968. He received the B.S. and M.S. degrees in Electronic Engineering from Yonsei University, Seoul, Korea, in 1991 and 1993, respectively. He received the Ph. D degree from the same university, in 1999. He has worked as a senior research engineer in the control system group at SAMSUNG Electronics Co., Korea, from 1999 to 2003. He is currently working as an Assistant Professor in the Department of Computer Engineering, Seokyeong University, Korea. His research interests include fuzzy control theory and applications, nonlinear analysis of intelligent control systems, adaptive and robust control, robotics and automation

Euntai Kim was born in Seoul, Korea, in 1970. He received the B.S. (summa cum laude) and the M.S. and the Ph.D. degrees in electronic engineering, all from Yonsei University, Seoul, Korea, in 1992, 1994 and 1999, respectively. From 1999 to 2002, he was a full-time lecturer in the Department of Control and Instrumentation Engineering at Hankyong National University, Kyonggi-do, Korea. Since 2002, he has joined the faculties of the School of Electrical and Electronic Engineering at Yonsei University, where he is currently an associate professor. His current research interests include computational intelligence and its application to intelligent service robot and intelligent home network. Dr. Kim is an associate editor of International Journal of Control and Systems.