Motion Planning Method Using Approximate Cell decomposition with Petri Nets for Single Mobile Robot

Alia K. Hassan **Moaid S. Murad** **Mizhur S. Alani**
University of Technology, Baghdad, Iraq

Abstract

The paper presents a method for single mobile robot motion planning. This method based on approximate cell decomposition motion planning approach for single mobile robot motion planning which require simple computation and it will find path from any start point to any goal position in the workspace if one exist. As in approximate cell decomposition we will divide the 2D configuration space into a set of reshaped cells of free space by using Petri net in which a marking of free position places is the result of the decomposition of the free space. An initial marking of free position places determine robot current position and goal marking determine the robot desired position finally Petri net reachability graph find a path from start position to the goal position. reachability graph is slightly different from connectivity graph.

طريقة لتحديد مسار الحركة لروبوت متحرك واحد باستخدام تجزئة الخلايا التقريبي و شبكات بتري

الخلاصة

في هذا البحث تم وضع طريقة لتحديد مسار الحركة لروبوت متحرك واحد هذه الطريقة تعتمد أساسا على طريقة هي تجزئة الخلايا التقريبي لتخطيط الحركة(approximate cell decomposition motion planning approach) لروبوت واحد وهي طريقة سهلة غير معقدة لإيجاد مسار حركة أن وجد من أي موقع ابتدائي(start position) إلى موقع الهدف (goal position). كما في طريقة تجزئة الخلايا التقريبي سوف يتم تجزئة فضاء الحركة ثنائي الأبعاد(two dimensional work space)إلى مجموعة من الخلايا ذات أشكال محددة لفضاء الحر(free space) باستخدام شبكات بتري والتي فيها التأشير(marking) للمواقع الحرة هو نتيجة التجزئة للفضاء الحر. التأشير الابتدائي يحدد موقع الروبوت الحالي والتأشير النهائي يحدد موقع الروبوت المراد . وباستخدام مخطط الوصول لشبكات بتري(Petri net reachability graph) يتم أيجاد طريق من أي موقع ابتدائي إلى موقع الهدف . مخطط الوصول هو يختلف تماما عن مخطط الربط(connectivity graph).

1-Introduction

A path planning application usually deals with an object to be moved and a workspace cluttered with obstacles for the robot moving from an initial configuration to a final one without colliding with any of the obstacles. There are many methods to robot path planning based on different workspace modeling approaches. Thus, each path planning method deals with a workspace modeling, i.e. an approximation of the workspace (Dragulescu *et al.*, 2001). In this paper the robot path planning moving in a two-dimensional workspace is solved by planning methods based on the cell decomposition approaches. In the cell decomposition methods after the workspace representation in the form of an image that is, depending on the necessities, more or less processed, the workspace is decomposed in area or volume subsets, named *cell*s. Depending on this decomposition, the path planning will be made using an exact cell decomposition method or an approximate cell decomposition method. As a result of applying these planning-methods there will be obtained an empty cell union, one of these cells is the initial cell $C_{initial}$ and another one the final cell C_{final}. Next, *connectivity graph* is constructed and searched; their nodes are cells of the free region and two nodes are connected if and only if the corresponding cells are adjacent, The outcome of the connectivity graph searching is a sequence of successive cells called *channel*; this channel connects the initial cell with the final one. The robot is moving along this channel (Dragulescu *et al.*, 1900). Our approach is based on exact cell decomposition method for single robot path planning which means guarantee that it will find a path from any start point to any goal position in the work space if one exist. Our method is Petri net motion planning method is developed with a C++ program.

2-Mobile Robot

Robot manipulators (first and for most the popular stationary robot arms) work fine for instance in assembly applications in factories. However, mobile robots offer some very important advantages, for instance:

Reach

Mobile robots are necessary if the problem the robot should solve is not restricted to some sufficiently small area.

Flexibility

If the position of the problem to be solved is not static, the mobile robot has the ability to pursue it (Johansson, 2000).

The basic problem of a mobile robot is that of *navigation* (motion planning): moving from one place to another by a coordination of planning, sensing and control. In any navigation scheme the desire is to reach a destination without getting lost or crashing into anything. Put simply the navigation problem is to find a path from start (S) to target (T) and traverse it without collision (Lazea ,2000).

3- Motion planning

The motion planning problem is the problem of finding a collision-free motion for one or more moving objects between the start and the goal configurations. A moving object can be a rigid object, or a jointed object such as an industrial manipulator. Motion planning has received much attention for the past two decades from robotics in order to automatically generate the movements of mobile robots and their arms, automatically plan and program the motions of manufacturing robots, and mechanical parts in assembling products (Chen and Hwang , 1998).

4-Motion Planning Approaches

To date, motion planning approaches can be classified into three categories 1) skeleton; 2) cell decomposition; 3) potential field. In the **skeleton** approach, the free space is represented by a network of one-dimensional (1-D) paths called a skeleton, and the solution is found by first moving the robot onto a point on the skeleton from the start configuration and from the goal, and connecting the two points via paths on the skeleton. Algorithms based on the visibility graph , the Voronoi diagram , and the silhouette (projection of obstacle boundaries) are examples of the skeleton approach. In the **cell-decomposition** approach the free space is represented as a union of cells, and a sequence of cells comprises a solution path. For efficiency, hierarchical trees, e.g, octree, are often used. In the **potential-field** approach, a scalar potential function that has high values near obstacles and the global minimum at the goal is constructed, and the robot moves in the direction of the negative gradient of the potential (Latombe 1991).

5-Configuration space

In his book, Robot Motion Planning (Latombe 1991), Latombe succinctly describes the notion (originally developed by Lozano-P’erez (Lozano and Wesley 1979) of a configuration space, C. The underlying concept is to represent the real-world robot as a point in an appropriate space, and to map obstacles into this same space. Then, the space contains a concise representation of the robot's geometrical constraints on motion, and a motion planner needs only to consider the path of the single point which represents the robot.

The configuration q of an object **A** specifies the exact position and orientation of **A** relative to a fixed reference frame. Therefore, the configuration space (often referred to as “C-space") of **A** is the set of all possible configurations of **A**. Obstacles are mapped into C-space by determining which configurations of the robot produce collisions with an obstacle; these configurations are deemed forbidden. Let **A(q)** denote the location of **A'**s particles when **A** is in configuratieon **q**. A C-space obstacle (or “C-obstacle") associated with a physical obstacle B is defined as

$$\mathbf{C\,B = \{\, q \in C \mid A(q) \cap B\ = \varnothing\}}$$

The complement of the C-obstacles is termed the frees pace CF where
CF = C\U CB. Motion plans are constructed in **CF** (Laubach and Lynn 1999). Configuration space transforms the problem of planning the motion of dimensioned object into the problem of planning the motion of a point. Configuration space **C** for the environment. The **CF** space which inside the C and not inside any of the configuration space obstacles **CB= {O^{i}},** i.e i$\in$ **[1, k],** for some k (Berg and Reveled 2000). All cell decomposition path planning methods are based on the same principle the decomposition of the work space in **CF** space which is not occupied and obstacle space **CB** which is occupied with obstacles (Johansson, 2000).

6- Approximate Cell Decomposition Motion planning

Cell decomposition methods are perhaps the motion planning methods which consist of decomposing the robot’s free space into simple regions, called cells, such that a path between any two configurations in a cell can be easily generated. A non directed graph representing relation between the cells is then constructed and searched. This graph is called the connectivity graph. Its nodes are the cells extracted from the free space and two nodes are connected by a link if and only if the two corresponding cells are adjacent. The outcome of the search is a sequence of cells called a channel. A continuous free path can be computed from this sequence. Cell decomposition is suitable since it generates polygonal paths which are suitable for the robot to move along straight lines. Approximate cell decomposition methods produce cells of predefined shape (e .g rectagloids) whose union is strictly included in the free space (Latombe,1991) this mean not use all the free space this can be described by the following function:

$$\cup_{i=1}^{m} cell_i \subset CF$$

As in approximate cell decomposition we will divide the 2D configuration space into a set of cells these cells have rectangle shape of free space by using Petri net in which a marking of free position places is the result of the decomposition of the free space in two decomposition either horizontally and vertically each representation represent either all the free space or sub set of it but the union of the two decomposition represent all the free space. An initial marking of free position places determine robot current position and goal marking determine the robot desired position finally Petri net reachability graph find a path from start position to the goal position. Reachability graph is slightly different from connectivity graph.

7-Petri Nets

Petri nets were proposed by the German mathematician Carl Adam Petri in the beginning of the 1960s, J Petri, 1962 K . Petri wanted to define a general-purpose graphical and mathematical model describing relations between conditions and events. The mathematical modeling ability of Petri nets makes it possible to set up state equations, algebraic equations, and other models governing the behavior of the modeled system. The graphical feature makes Petri nets suitable for visualization and simulation. PN can be defined as a 4-tuple N={ P, T, I, O} where:

· *P ={P1,P2, ...Pn } is a finite, nonempty set of places.*
· *T ={ T1,T2, ..., Tm }is a finite, nonempty set of transitions.*
· *$P \cap T=\varnothing$,i.e., the sets P and T are disjoint.*
· *I : PXT $\rightarrow$ {0,1} is the input incidence function.*
· *O : P X T$\rightarrow$ { 0,1} is the output incidence function.*

A marked PN is a pair PN=(N,M0) in which N is an unmarked N and M0 is the initial marking.I(Pi, Tj) is the weight of the arc connecting place Pi with transition Tj .This weight is 1 if the arc exists and 0 if not. O(Pi, Tj) is the weight of the arc connecting transition Tj with place Pi. This weight is 1 if the arc exists and 0 if not Petri net can be described by (m x n) matrix called incidence matrix(A) where m is the transitions and n represent places it entries –1 represent input to transition and 1 output from transition and 0 if there is no relation between place and transition (Laubach and Lynn ,1999; Murata , 1989) .

Example 1:

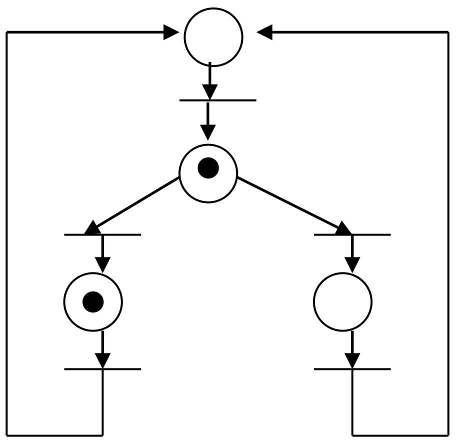

Figure 1 Petri net graph 0)

P =< P1,P2,P2,P4 > T=<T1,T2,T3,T4,T5 >

I(t1)={p1}	O(t1)={p2}
I(t2)={p2}	O(t2)={p3}
I(t3)={p2}	O(t3)={p4}
I(t4)={p3}	O(t4)={p1}
I(t5)={p4}	O(t5)={p1}

$$M0=\begin{array}{cccc} P1 & P2 & P3 & P4 \\ (0 & 1 & 1 & 0) \end{array}$$

$$A=\left(\begin{array}{ccccc} & p1 & p2 & p3 & p4 \\ t1 & -1 & 1 & 0 & 0 \\ t2 & 0 & -1 & 1 & 0 \\ t3 & 0 & -1 & 0 & 1 \\ t4 & 1 & 0 & 0 & 0 \\ t4 & 1 & 0 & -1 & -1 \end{array}\right)$$

7-1 Reachability graph

Reachability is a fundamental property for studying the behavior of the system modeled by a Petri Net. Let p and t denotes the number of places and transitions in a Petri Net (PN), respectively. A *marking or state vector Mk* of PN is a (p x 1) column vector of nonnegative integers . The j th entry in Mk (j = 1, 2, 3,...., p) assigns the number of tokens in the j th place at the given state. Specifically, M0 denotes the *initial marking (initial condition-state*).

A marking Mn is said to be *reachable* from an initial marking M0, if there exist a firing sequence, which transforms M0 to Mn. A *firing or occurrence sequence* is denoted by σ= M0 t1 M1 t2 M2 t3 …… tn Mn or simply σ = t1 t2 t3 … tn (where ti , i = 1,2,…..,n denotes the i th transition). In this case, Mn is reachable from M0 by σ and is written as M0 [σ >Mn. The set of all possible markings reachable from M0 in a PN is denoted by R(PN, M0) or simply R(M0) and is called as the *reachability se*t. The *reachability problem* of Petri Nets is the problem of finding if Mn∈R(M0) for a given marking Mn in a PN. The reachability set of a PN can be represented by a tree called *reachability tree* whose nodes are the markings of the PN under consideration and whose arcs represent the possible changes in state resulting from the firing of transitions (Murata , 1989) .

8- The Proposed Method

Petri net (PN) path planning method deals with the two dimension work space modeled with PN . In our path planning method the set of places P is formed by positions of the robot and the set of transitions T consists of the robot movement such that a transition $t_q = t_{i-j}$ drives the robot from place P_i to place P_j . The weighting of arc assigns value 1 to each arc linking places and transitions as each transition determines the movement of the robot between two places. The net marking depend on the place where the robot is situated. There for the firing of the transition t_{i-j} depends on the position of the robot.PN planning method decompose configuration space **C** into free space **CF** which is the result of the intersection between the left to right movement free regions **Clr** and the up to down movement free regions **Cud** .while skipping the obstacles regions **CB**. If there is only one **CF** and both start position((Ps) and goal position (Pg) belong to it then we guarantee to find a path . If there is more than one **CF's** and each of Ps and Pg belonging each one in a different **CF's** we guarantee there is no path to find. PN planning method consists of five steps are:

<u>Step1</u>

Input: bit map file

Output: character matrix of 0's and 1's

The first step has an input data a monochrome raster image file (bit map file) representing the environment . this bit map file data converted into a character matrix of 0's and 1'sto represent the environment to be processed next.

<u>Step2</u>
Input: character matrix of 0’s and 1’s
Output: PN incidence metrics Clr and Cud.
In this step the PN representation for the environment will be made in a two metrics. First one Clr for the left to right movement and the other Cud for up to down movement.
<u>Step3:</u>
Input: PN incidence metrics Clr and Cud.
Output :set of free regions CLR and CUD.
Now for each matrix construct the free space regions one for the left to right movement CLR and the other for up to down movement CUD and CB which represent the zero column.
<u>Step4:</u>
Input: set of free regions CLR and CUD.
Output: contiguous free regions CF’s
The intersection of the sets of free regions CLR and CUD will produce a contiguous free regions CF’s for possible movement in two directions.
<u>Step5:</u>
Input: CF’s, start position Ps and goal position Pg
Output: collision free paths
<u>Note</u>
From the CF’s we can decide if there is a path:
If there is only one CF and both Ps and Pg belong to it then we guarantee to find a path. If there is more than one CF’s and each of Ps and Pg belonging each one in a different CF’s we guarantee there is no path to find.

9- Results

In this section show by examples how proposed method can find a path for a robot move first in an environment with one obstacle and one free space region then an environment with three obstacle and on free region and with one obstacle and two free regions .

Exmple1

Ps	P2	P3	P4	P5
P6	P7	■	P9	P10
P11	■	■	■	P15
■	■	■	P19	P20
P21	■	P23	P24	P25
P26	P27	P28	P29	Pg

CLR	CUD	CB
P1,P2,P3,P4,P5	P1,P6,P11	P8,P12,P133,P14,P15,16,P17,P18, P22
P6,P7	P2,P7	
P9,P10	P4,P9	

P19,P20	P5,P10,P15,P20,P25,P30	
P23,P24,P25	P21,P26	
P26,P27,P28,P29,P30	P23,P28	
	P19,P24,P29	

CF= CLR ∩ CU
={P1,P2,P3,P4,P5,P6,P7,P9,P10,P11,P15,P19,P20,P21,P23,P24,P25,P26,P27,P28,P29,P30}

Thus the free space constructed from one single region and both start position P1 and goal position P30 are in this single free region so we can find the collision free path in between by using reachabilty graph Initial marking is
M0=(1, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0), and
Goal marking is
Mg=(0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1)
There is two possible paths reachable from M0 to Mg throng the following firing sequence

σ1=(t1lr , t2lr ,t3lr , t6ud , t5lr , t10ud , t11ud , t12ud , t13ud)
σ2=(t1lr , t2lr ,t3lr , t4l r, t9ud, t10ud , t11ud , t12ud , t13ud)

Note *each represent a transition (legal movement either up to down tud or left to right tlr) its sequence depend on the incidence matrix which represent the free space*

Example2

P1	P2	P3	P4	P5
P6	P7	■	P9	P10
Ps	■	■	■	P15
■	■	■	■	■
P21	P22	P23	P24	Pg

CLR	CUD	CB
P1,P2,P3,P4,P5	P1,P6,P11	P8,P12,P13,P14,P16,P17,P18,P19,P20
P6,P7	P2,P7	
P9,P10	P4,P9	
P21,P22,P23,P24,P25	P5,P10,P15	

CF=CLR ⋂ CUD =(CF1,CF2)=({P1,P2,P3,P4,P5,P6,P7,P9,P10,P11,P15} , {P21,P22,P23,P24,P25})

Ps ∈ CF1 and Pg ∈ CF2 in this case each start position and goal position are in different free region of free space .

Example3

P1	P2	P3	P4	P5
P6	P7	P8	P9	P10
Ps	P12	⬛	P14	P15
P16	P17	P18	⬛	P20
P21	⬛	P23	P24	Pg

CLR	CUD	CB
P1,P2,P3,P4,P5	P1,P6,P11,P16,P21	P13,P19,P22
P6,P7,P8,P9,P10	P2,P7,P12,P17	
P11,P12	P3,P8	
P14,P15	P18,P23	
P16,P17,P18	P4,P9,P14	
P23,P24,P25	P5,P10,P15,P20,P25	

CF=CLR $\cap$ CUD=
{p1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P14,P15,P16,P17,P18,P20,P21,P23,P24,P25}

Both Ps and Pg are in a single free space so we can find a path so the initail marking and goal marking are:
M0=(0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
Mg=(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1)
Possible paths are
σ = (t10lr , t11lr , t11l r,t9ud , t12lr , t13lr)
σ = (t3ud ,t7ud , t11l r, t9ud , t12lr , t13lr)

10-Implementation

This method is implemented with C++ program then with matlab programming and we compare between the two implementation .

Table -1-

Example number	Start position cell	Goal position cell	Required time in c++ simulation in seconds	Required time in matlab simulation in seconds
1	1	30	0.03900	1.7600
2	11	25	0.05200	0.8200

3	1	25	0.04400	1.5900

From the above table (table -1) show that C++ implementation require time less than in matlab implementation.

11-Conclusions

A proposed method for single robot motion planning is described. The presented method uses the PN for single mobile robot motion planning. PN is used for modeling the workspace and control robot movement in this workspace. Finding path for the robot motion is guaranteed by using this method if there is a path since this method make exact representation by using two approximate decomposition for the environment by two approximate representation of PN sub nets and the PN reachability graph will produce the shortest path. C++ implementation for this method require less time than matlab implementation.

References

Berg M., and Reveled M., (2000). ”Computational Geometry: algorithms and applications”, Springer Verlag Berlin Hiedlberg New York.

Chen C., and Hwang K., (1998). *Member, IEEE*, “SANDROS: A Dynamic Graph Search Algorithm for Motion Planning”, IEEE TRANSACTIONS ON Robotics and Automation, vol. 14, no. 3, June.

Dragulescu D., Toth-Tascau M., and. Dragomir L, (2001). ”Motion Planning using approximate Cell Decomposition Method” , Faculty of mechanical Engineering , Bd. Mihai Viteazul No.1,1900. Timisora(ROMANIA), *http//www.robotics.ucv.ro/wmro2001/capture.html.*

Johansson C., (1999). “A graphical language for batch control”, PhD Thesis, Lund Institute of Technology.

Johansson R., (2000). “ *Intelligent Motion Planning for a Multi-Robot System*” ,Master’s thesis in Computer Science at the school of Computer Science and Engineering, Royal Institute of technology year.

Latombe J. claud, (1991). “Robot motion planning”, New York : Klwer .

Laubach S. Lynn, (1999). "Theory and Experiments in Autonomous Sensor-Based Motion Planning with Applications for Flight Planetary, Microrovers", Thesis, California Institute of Technology Pasadena, California.

Lazea G., (2000). “Aspects on path planning for mobile robots “, Technical University of Cluj-Napoca Automation Department, *http>//www.users.uctluj.ro/`emlupu/buchrest.pdf.*

Lozano T. Perez and Wesley M.,(1979). "An algorithm for planning collision-free paths among polyhedral obstacles", Commun .ACM, Vol. 22, no ,10, pp. 560-570.

Murata T., (1989). ‘Petri nets Properties, Analysis and Applications’, proceeding of IEEE, vol 77, no .4 April.