World Academy of Science, Engineering and Technology
International Journal of Aerospace and Mechanical Engineering
Vol:2, No:5, 2008

A 7DOF Manipulator Control in an Unknown Environment based on an Exact Algorithm

Pavel K. Lopatin, and Artyom S. Yegorov

***Abstract*—**An exact algorithm for a n-link manipulator movement amidst arbitrary unknown static obstacles is presented. The algorithm guarantees the reaching of a target configuration of the manipulator in a finite number of steps. The algorithm is reduced to a finite number of calls of a subroutine for planning a trajectory in the presence of known forbidden states. The polynomial approximation algorithm which is used as the subroutine is presented. The results of the exact algorithm implementation for the control of a seven link (7 degrees of freedom, 7DOF) manipulator are given.

***Keywords*—**Manipulator, trajectory planning, unknown obstacles.

I. Introduction

IN contemporary society robots and manipulators are used in different spheres of life. Robot should be as autonomous as possible and should effectively operate in a natural environment.

In the beginning of the robotic era, robots operated in a workspace which was free of obstacles. Later the works dedicated to the invention of algorithms for the control of robots in the presence of obstacles began to appear. There are algorithms which guarantee finding a trajectory in the presence of known obstacles, if such trajectory exists [1, 4, 10]. Some authors use the artificial potential methods (see, for example, [2]). In this method a robot is represented by a point, regions in the workspace that are to be avoided are modeled by repulsive potentials and the region to which the robot is to move is modeled by an attractive potential. In a general situation there is no guarantee that a collision-free path will always be found, if one exists [1]. There are different graph searching methods [10] which find the trajectory avoiding obstacles (even an optimal one), if it exists. It is easier to use such methods in the case where we have full information about free and forbidden points before the beginning of the movement. A computer may then calculate a preliminary trajectory and after that the manipulator may realize this trajectory. But in case of unknown obstacles the manipulator has to investigate its environment and plan its trajectory simultaneously. Then the difficulty arises that graph algorithms of route searching demand breadth searching, otherwise reaching the target configuration will not be guaranteed. But during breadth searching it is necessary to switch from one node **q** to another node **q*** which may be not adjacent. Then the problem of the manipulator moving from **q** to **q*** arises, and as a result the total sum of the manipulator's movements becomes very big [3]. It is also known that the "depth-first" algorithms do not guarantee reaching the goal [3].

Attempts of creating algorithms for robot control in presence of unknown obstacles were made. Most of them cover various two-dimensional cases [9].

In [9] the algorithm for the control of manipulators in the presence of unknown obstacles in three-dimensional space is given. Though this algorithm guarantees reaching a target position, it has such a limitation that the manipulator should not have more than three degrees of freedom.

In [11] the n-dimensional case is considered. The algorithm is based on the solution of the system of nonlinear equations using the Newton method and therefore it cannot guarantee reaching a target position.

In [4] algorithms for moving a robot in the presence of uncertainty (including cases of unknown environment) are considered. The algorithms are based on the sequential decision theory. In general case the algorithms do not guarantee reaching the goal. In cases when the algorithms use searching on a graph the above mentioned difficulty arises connected with multiple mechanical movements.

II. Task Formulation and the Exact Algorithm

A. Preliminary Information

We will consider manipulators which consist of n rigid bodies (called links) connected in series by either revolute or prismatic joints [5]. We must take into account that because of manipulator's constructive limitations the resulting trajectory **q**(t) must satisfy the set of inequalities

$$\mathbf{a}^1 \le \mathbf{q}(t) \le \mathbf{a}^2 \qquad (1)$$

for every time moment, where $\mathbf{a}^1$ is the vector of lower limitations on the values of generalized coordinates comprising **q**(t), and $\mathbf{a}^2$ is the vector of higher limitations.

The points satisfying the inequalities (1) comprise a hyperparallelepiped in the generalized coordinate space. We will consider all points in the generalized coordinate space which do not satisfy the inequalities (1) as forbidden.

We will have to move the manipulator from a start configuration $\mathbf{q}^0=(q_1^0, q_2^0, \dots, q_n^0)$ to a target configuration $\mathbf{q}^T=(q_1^T, q_2^T, \dots, q_n^T)$. In our case the manipulator will have to move amidst unknown obstacles. If in a configuration **q** the manipulator has at least one common point with any obstacle then the point **q** in the configuration space will be considered as forbidden. If the manipulator in **q** has no common points with any obstacle then the **q** will be considered as allowed.

So, in our problem a manipulator will be represented as a point which will have to move in the hyperparallelepiped

Manuscript received April 17, 2008. This work was supported by the Russian Foundation for Basic Research, grant №05-08-01199a.

P. K. Lopatin is with the Department of Informatics and Computing Techniques, Siberian State aerospace university named after academician M.F.Reshetnev, Krasnoyarsk, 660014, Russia (phone: (7)(3912)91-92-41, e-mail: pavel-lopatin@yandex.ru).

A. S. Yegorov is with the Department of System Analysis, Siberian State aerospace university named after academician M.F.Reshetnev, Krasnoyarsk, 660014, Russia (e-mail: earts@yandex.ru).

from $\mathbf{q}^0$ to $\mathbf{q}^T$ and the trajectory of this point should not intersect with the forbidden points.

B. Preliminary Considerations

Let us make the following considerations:

1) The disposition, shapes and dimensions of the obstacles do not change during the whole period of the manipulator movement.

2) It is known in advance, that the target configuration is reachable (that is, we know that in the generalized coordinates space it is possible to find a line connecting $\mathbf{q}^0$ и $\mathbf{q}^T$, and that this line will not intersect with any forbidden point).

3) The manipulator has a sensor system which allows to define whether the manipulator intersects with an obstacle or not for an arbitrary current configuration $\mathbf{q}$ and for all configurations lying in a small r-neighborhood of $\mathbf{q}$. r-neighborhood is a hyperball in the generalized coordinates space with the center in $\mathbf{q}$ and with a radius r>0. We will not consider the structure of the sensor system.

4) We denote a set of all configurations from a r-neighborhood of $\mathbf{q}$ as Y($\mathbf{q}$). The set of all forbidden points from Y($\mathbf{q}$) will be denoted as Q($\mathbf{q}$), the set of all permitted points from Y($\mathbf{q}$) will be denoted as Z($\mathbf{q}$). An r-neighborhood of $\mathbf{q}$ with the sets Z($\mathbf{q}$) and Q($\mathbf{q}$) may look as follows (Fig. 1). Note that the sets Z($\mathbf{q}$) and Q($\mathbf{q}$) may be not continuous.

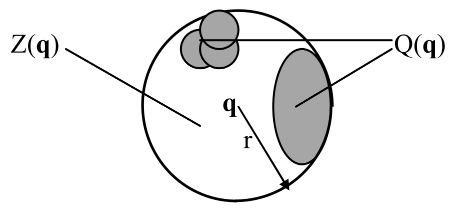

Fig. 1 An example of the r-neighborhood

The considerations 1)-4) cover wide range of manipulators' applications.

C. The Exact Algorithm for Manipulators' Control in the Unknown Environment

Consider the exact algorithm for manipulators' control in the unknown environment [6]. We will denote the points where generation of a new trajectory occurs as $\mathbf{q}^n$, n=0, 1, 2,… We will call such points "trajectory changing points". Before the algorithm work n=0 and $\mathbf{q}^n = \mathbf{q}^0$.

STEP 1. The manipulator is in a point $\mathbf{q}^n$, n=0, 1, 2,…, and its sensor system supplies information about the r-neighborhood of the $\mathbf{q}^n$. After that the manipulator generates in the configuration space a preliminary trajectory L($\mathbf{q}^n$, $\mathbf{q}^T$). The L($\mathbf{q}^n$, $\mathbf{q}^T$) should satisfy the following conditions: I) connect the $\mathbf{q}^n$ and the $\mathbf{q}^T$; II) no point from the L($\mathbf{q}^n$, $\mathbf{q}^T$) coincides with any point from the sets $\bigcup_{i=0}^{n} Q(\mathbf{q}^i)$, in other words the preliminary trajectory should not intersect with any known forbidden point; III) satisfy the conditions (1).

The manipulator starts to follow the L($\mathbf{q}^n$, $\mathbf{q}^T$). The algorithm goes to STEP 2.

STEP 2. While following the L($\mathbf{q}^n$, $\mathbf{q}^T$) two results may happen:

а) the manipulator will not meet forbidden points unknown earlier and therefore will reach the $\mathbf{q}^T$. Upon the reaching of the $\mathbf{q}^T$ the algorithm terminates its work;

в) the manipulator will come to such a point (making at first operation n=n+1, let us define it $\mathbf{q}^n$, n=1,2,…), that the next point of the preliminary trajectory is forbidden. The algorithm goes to STEP 1.

D. Theorem

If the manipulator moves according to the exact algorithm it will reach the target configuration in a finite number of steps.

Proof is given in [6].

III. Using the Polynomial Approximation Algorithm as a Subroutine in the Exact Algorithm

A. Reducing the Exact Algorithm to a Finite Number Calls of a Subroutine for a Trajectory Planning in Known Environment

Every time when the manipulator generates a new trajectory according to the STEP 1 of the exact algorithm, two cases may happen: either the manipulator will not meet an obstacle and therefore it will reach the $\mathbf{q}^T$ in a finite number of steps (because the length of the trajectory is finite) or the manipulator will meet an unknown obstacle and will have to plan a new trajectory. In [6] it is proved that the number of cases when the manipulator will have to plan a new trajectory according to the STEP 1 will be finite. Therefore, in the exact algorithm the problem of a manipulator control in the presence of unknown obstacles is reduced to the solution of a finite number tasks of trajectory planning in the presence of known forbidden states. In other words, the exact algorithm will make a finite number of calls of a subroutine which will solve the problem stated in STEP 1. In the rest of the article we will call this subroutine the SUBROUTINE. We took the polynomial approximation algorithm as algorithm for the SUBROUTINE.

B. Polynomial Approximation Algorithm

Denote the components of vector-function $\mathbf{q}(t)$ as q_1, q_2, …, q^n. Write down the restrictions using new variables:

$$q_j(0) = q_j^0, \quad q_j(1) = q_j^T, \quad j = \overline{1,n}. \tag{2}$$

$$q_j^L \le q_j(t) \le q_j^H, \quad j = \overline{1,n}. \tag{3}$$

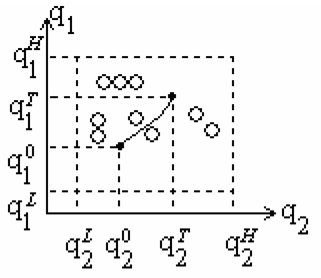

Fig. 2 Configuration space

Specify the obstacles by hyperspheres with centers $(q_{m1}^p, q_{m2}^p, \dots, q_{mn}^p)$ and radius $r = \frac{\Delta q}{2} \cdot 1.1$, where q_{mi}^p correspond to the components of vectors $\mathbf{p}_m$, m = 1, 2, …, M, Δq is step of configuration space discretization. The value of the radius *r* is chosen so that if two obstacles are located on neighbor nodes of the grid and their centers differ only in one component, the corresponding hyperspheres intersect. Otherwise the hyperspheres don't intersect. Then the requirement of obstacles avoiding trajectory can be written as follow:

$$\sum_{i=1}^{n}(q_i(t)-q_{mi}^p)^2 \geq r^2 \quad \forall t \in [0;1],\ m=1,2,\ldots,M. \qquad (4)$$

The left part of this inequality is squared distance between the trajectory point at moment *t* and the center of the *m*-th obstacle.

We will search the trajectory in the form of polynomials of some order *s*:

$$q_j(t)=\sum_{i=0}^{s}c_{ji}t^i, \quad j=\overline{1,n} \qquad (5)$$

Here c_{ji} are unknown coefficients.

Substitute $t = 0$ and $t = 1$ in (5):

$$q_j(0)=c_{j0}+c_{j1}\cdot 0+c_{j2}\cdot 0^2+\cdots+c_{js}\cdot 0^s=c_{j0}$$

$$q_j(1)=c_{j0}+c_{j1}\cdot 1+c_{j2}\cdot 1^2+\cdots+c_{js}\cdot 1^s=c_{j0}+\sum_{i=1}^{s}c_{ji}$$

$$j=\overline{1,n}$$

Taking into account requirements (2):

$$c_{j0}=q_j^0. \qquad (6)$$

$$\sum_{i=1}^{s}c_{ji}=q_j^T-q_j^0. \qquad (7)$$

$$j=\overline{1,n}$$

Divide duration [0; 1] into *K*+1 pieces by *K* intermediate points $t_1, t_2, \ldots, t_K$. Then the requirements (3) and (4) will be as follow:

$$\sum_{i=0}^{s}c_{ji}t_k^i \geq q_j^L, \quad j=\overline{1,n}$$

$$\sum_{i=0}^{s}c_{ji}t_k^i \leq q_j^H, \quad j=\overline{1,n} \qquad (8)$$

$$\sum_{j=1}^{n}\Big(\sum_{i=0}^{s}c_{ji}t_k^i-q_{mj}^p\Big)^2 \geq r^2,$$

$$m=1,2,\ldots,M, \quad k=1,2,\ldots,K.$$

Thus, it is necessary to find such coefficients c_{ji} ($j=\overline{1,n}$, $i=\overline{1,s}$) which satisfy the system of equations (6), (7) and inequalities (8). Obviously, the coefficients c_{j0} are easily found from the equations (6). Express the coefficients c_{js} from the equations (7):

$$c_{js}=q_j^T-q_j^0-\sum_{i=1}^{s-1}c_{ji}, \quad j=\overline{1,n}.$$

Substitute c_{j0} and c_{js} in (8):

$$q_j^0+\sum_{i=1}^{s-1}c_{ji}t_k^i+(q_j^T-q_j^0-\sum_{i=1}^{s-1}c_{ji})t_k^s \geq q_j^L,$$

$$q_j^0+\sum_{i=1}^{s-1}c_{ji}t_k^i+(q_j^T-q_j^0-\sum_{i=1}^{s-1}c_{ji})t_k^s \leq q_j^H, \quad j=\overline{1,n}, \qquad (9)$$

$$\sum_{j=1}^{n}\left[q_j^0+\sum_{i=1}^{s-1}c_{ji}t_k^i+(q_j^T-q_j^0-\sum_{i=1}^{s-1}c_{ji})t_k^s-q_{mj}^p\right]^2 \geq r^2,$$

$$m=1,2,\ldots,M, \quad k=1,2,\ldots,K.$$

Thus, it is necessary to solve the system of $(M + 2n)\cdot K$ nonlinear inequalities. This problem can be replaced by the problem of optimization of some function *F*(*C*):

$$F(C)=\begin{cases}E(C), & \text{if } E(C)<0\\ P(C), & \text{if } E(C)=0,\end{cases}$$

where *C* is vector of coefficients ($c_{1,1}, c_{1,2}, \ldots, c_{1,s-1}, c_{2,1}, c_{2,2}, \ldots, c_{2,s-1}, \ldots, c_{n,1}, c_{n,2}, \ldots, c_{n,s-1}$), *E*(*C*) is measure of restrictions violation, *P*(*C*) is estimated probability that trajectory not intersects with unknown obstacles.

Let's define function *F*(*C*). First, introduce function defining the trajectory point for the vector of coefficients *C* at the moment *t*:

$$L_j(C,t)=q_j^0+\sum_{i=1}^{s-1}c_{ji}t^i+(q_j^T-q_j^0-\sum_{i=1}^{s-1}c_{ji})t^s.$$

The following functions correspond to the inequalities of the system (8), they show the violation of upper and lower bounds and the intersection with the obstacles of trajectory at the moment *t*:

$$E_L(C,t)=\sum_{j=1}^{n}I(L_j(C,t)-q_j^L),$$

$$E_H(C,t)=\sum_{j=1}^{n}I(q_j^H-L_j(C,t)),$$

$$E_P(C,t)=\sum_{m=1}^{M}I\left[\sum_{j=1}^{n}(L_j(C,t)-q_{mj}^p)^2-r^2\right],$$

$$I(z)=\begin{cases}z, & \text{if } z<0,\\ 0, & \text{if } z\geq 0.\end{cases}$$

Because of using of the operator *I*, the items in the above functions are negative only if corresponding restrictions are violated. The more violation of the restriction, the more the value of the function. If particular restriction is not violated, the corresponding function element is zero. In any case, values of functions can't be positive.

Join all restrictions into single function (taking into account all discrete moments except $t = 0$ and $t = 1$):

$$E(C)=\sum_{k=1}^{K}\left[E_L(C,t_k)+E_H(C,t_k)+E_P(C,t_k)\right].$$

Thus, *E*(*C*) takes negative values if at least one restriction is violated and becomes zero otherwise, that is if the vector *C* satisfies all the inequalities of the system (9).

The function *P*(*C*) was introduced to make possible comparison of trajectories which not violate the restrictions. Assume $p(d) = e^{-2d}$ is probability that there is an unknown obstacle in some point, where *d* is distance between this point and the nearest known obstacle. Then $P(C)=\prod_{m=1}^{M^*}p(D(C,O_m))$, where $\{O_1, O_2, \ldots, O_{M^*}\}$ is set of obstacles along which the trajectory lies, $D(C,O) = \min_k \sqrt{\sum_{j=1}^{n} (L_j(C,t_k) - O_j)^2 - r^2}$ is distance between the trajectory C and the obstacle O. A trajectory *lies along* an obstacle O if such point of trajectory $L(t_k)$ exists as an obstacle O is nearest among all obstacles for this point. Usage of function P(C) would promote the algorithm to produce trajectories which lie far from unknown obstacles.

Since function $F(C)$ is multiextremal, the genetic algorithm is used to find the desired vector.

C. Optimization of the Restriction Function using Genetic Algorithm

Genetic algorithm is based on collective training process inside the population of individuals, each representing search space point. In our case search space is space of vectors C. Encoding of vector in the individual is made as follows. Each vector component is assigned the interval of values possessed by this component. The interval is divided into a quantity of discrete points. Thus, each vector component value is associated with corresponding point index. The sequence of these indexes made up the individual.

The algorithm scheme:

1. Generate an initial population randomly. The size of population is N.
2. Calculate fitness values of the individuals.
3. Select individuals to the intermediate population.
4. With probability P_C perform crossover of two individuals randomly chosen from the intermediate population and put it into new population; and with probability $1 - P_C$ perform reproduction – copy the individual randomly chosen from intermediate population to new population.
5. If size of new population is less than N, go to Step 4.
6. With given mutation probability P_M invert each bit of each individual from new population.
7. If required number of generations is not reached go to Step 2.

The fitness function matches $F(C)$. On the third step the tournament selection is used: during the selection the N tournaments are carried out among m randomly chosen individuals (m is called tournament size). In every tournament the best individual is chosen to be put into the new population.

On the fourth step the arithmetical crossover is used. The components of offspring are calculated as arithmetic mean of corresponding components of two parents.

D. Quantization of the Path

After the polynomials coefficients specifying the route are found, it's necessary to get the sequence of route discrete points $\mathbf{q}^0, \mathbf{q}^1, \ldots, \mathbf{q}^T$. The first point ($\mathbf{q}^0$) is obtained from the initial values. The rest of points can be found using the following algorithm:

1. $t = 0$; $i = 0$.
2. $t_H = 1$.
3. $t^* = \dfrac{t + t_H}{2}$.
4. Find the point $\mathbf{q}^*$ with coordinates $(q_1^*, q_2^*, \ldots, q_n^*)$ in whose neighborhood the trajectory is lying at the moment t^*: $q_j^* - \dfrac{\Delta q}{2} \le q_j(t^*) < q_j^* + \dfrac{\Delta q}{2} \quad \forall j = \overline{1,n}$
5. If $\mathbf{q}^*$ equals $\mathbf{q}^i$, then $t = t^*$, go to Step 3.
6. If $\mathbf{q}^*$ is not a neighbor of $\mathbf{q}^i$, then $t_H = t^*$, go to Step 3.
7. If $\mathbf{q}^*$ is not forbidden, then go to Step 9.
8. If $q_{ij} - \Delta q \le q_j(t^*) \le q_{ij} + \Delta q \quad \forall j = \overline{1,n}$, where $\mathbf{q}^{\mathbf{i}}_{\mathrm{j}}$ are the coordinates of $\mathbf{q}^i$, then $t = t^*$, otherwise $t_H = t^*$, go to Step 3.
9. $i = i + 1$; $\mathbf{q}^i = \mathbf{q}^*$.
10. If $\mathbf{q}^i = \mathbf{q}^T$, then the algorithm is finished, otherwise go to Step 2.

IV. Experimental Results

Consider the following experimental set (Fig. 4). It is necessary to move a seven-link manipulator (Fig. 3) from the start configuration $\mathbf{q}^0 = (1.57; 1.57; 0; 4.71; 0; 4.71; 0)$ (rad) to the target configuration $\mathbf{q}^T = (4.71; 1.57; 0; 0; 0; 0; 0)$(rad). There are the following limitations on the generalized coordinates: $0 \le q_i(t) \le 6.28$, $i = 1,2,\ldots,7$. The length of each link is 10 points. There are four cuboid obstacles in the working area. Each obstacle is described by six values: the length, width, height and the coordinates of the origins attached to the obstacles in the basic coordinate system (Table I).

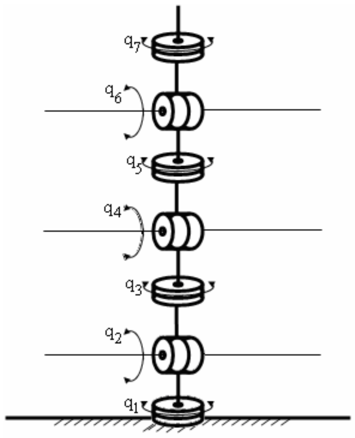

Fig. 3 The manipulator kinematic scheme

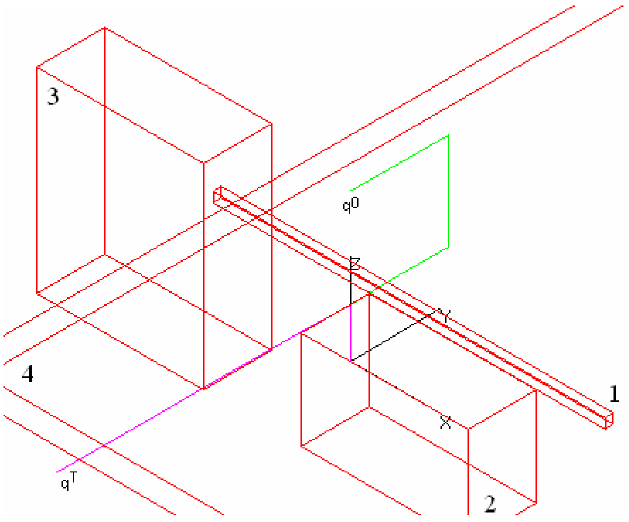

Fig. 4 Experimental set

TABLE I
THE CHARACTERISTICS OF OBSTACLES

№	*x*	*y*	*z*	Length	width	height
1	-30	2	12	80	1.6	2
2	10	-20	0	34	14	20
3	-44	-20	0	34	14	40
4	-40	-40	-10	200	200	10

The parameters of algorithms are as follow:
1. Polynomial order *s*: 10.
2. Number of time pieces *K*: 100.
3. Polynomials coefficients precision: 10^{-5}.
4. Population size *N*: 20.
5. Number of generations: 20.
6. Probability of crossover P_C: 0,5.
7. Probability of mutation P_M: 0,1.
8. Tournament size *m*: 5.

The working time of the exact algorithm depending on different *number_of_discretes* is given in the Table II. The Δq is calculated as the difference between upper and lower bounds of $\mathbf{q}(t)$ (that is 6.28) divided on the *number_of_discretes*. The working time is a sum of three elements: trajectory search time, time to check whether trajectory intersects with unknown obstacles, manipulator moving time (12° per second).

TABLE II
EXPERIMENTAL RESULTS

Obstacles	*number_of_ discretes*	Working time, seconds
1, 2	40	58
	60	82
	120	153
	240	150
	360	125
1, 2, 3	40	83
	60	96
	120	154
	240	233
	360	251
1, 2, 3, 4	40	233
	60	340
	120	761
	240	1142
	360	1169

The tests were done on the processor AMD Athlon XP 1800+ (1533 MHz).

The key steps of manipulator trajectory for the last test case (with four obstacles) are shown on the Fig. 5.

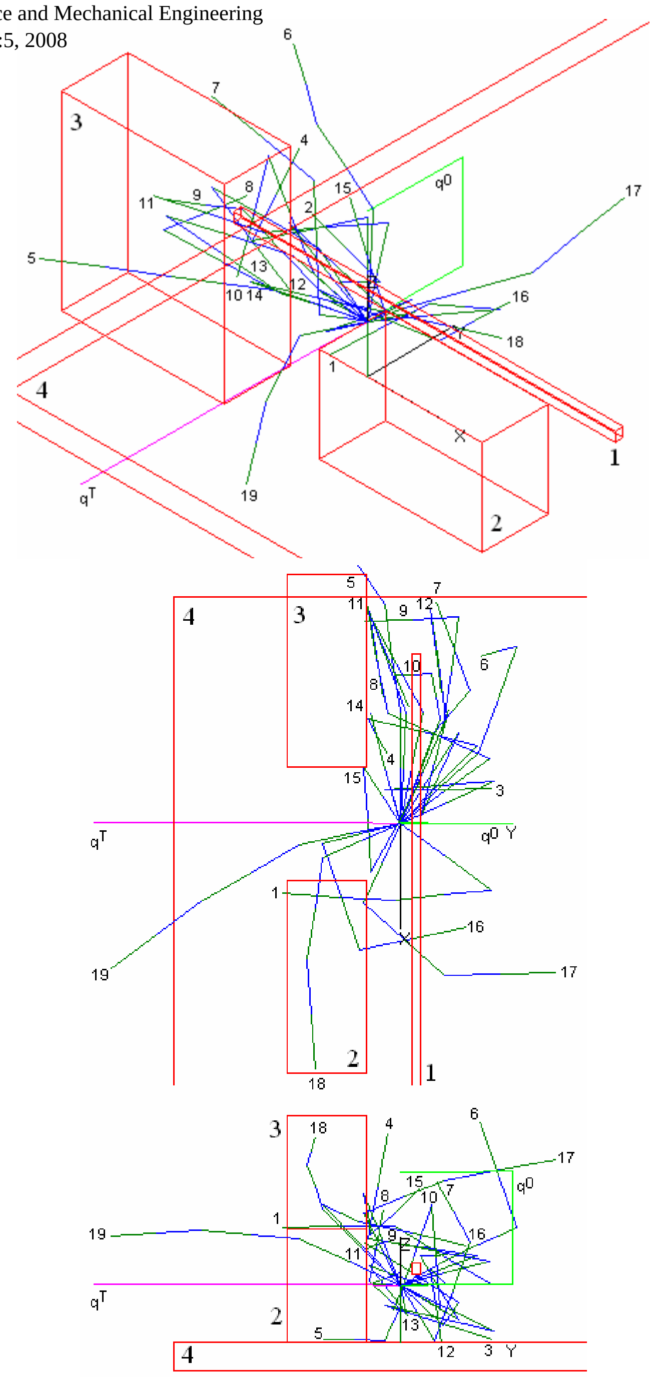

Fig. 5 The key steps of manipulator movement trajectory

V. CONCLUSION

The exact algorithm for a n-link manipulator movement amidst arbitrary unknown static obstacles was presented. The algorithm guarantees the reaching of a target configuration in a finite number of steps. The algorithm is reduced to a finite number of calls of a subroutine for planning a trajectory in the presence of known forbidden states. The polynomial approximation algorithm which is used as the subroutine was presented. The results of the exact algorithm implementation for the control of a seven link manipulator are given.

During experiments the following advantages and disadvantages of the polynomial approximation algorithm were discovered:
1. Algorithm is applicable to the n-dimensional space.
2. Algorithm works well if a searched path is not too «snaky» curve, i.e. when the obstacles are grouped in

big blocks and there are not many such blocks. If the configuration space is strongly encumbered, a path may be not found.

3. The quality of the algorithm's work depends on the chosen degree of the polynomial and the value of time discretes. Making these values bigger leads, from one side, to a possibility of finding complex trajectories, but from another side – makes the search time bigger.
4. If the robot, following the generated path, meets with unknown obstacle, it will be necessary to carry out the work of a trajectory finding from the very beginning, i.e. the previous preliminary trajectory is not used.

Here are the criteria which should be satisfied by an algorithm for the subroutine:

- It should be applicable to the n-dimensional case;
- It should guarantee finding a path in the presence of known forbidden states;
- In case of new call of the SUBROUTINE should be done the minimum work for finding a path in the presence of known forbidden states.

REFERENCES

[1] C. Ahrikhencheikh, A. Seireg, *Optimized-Motion Planning: Theory and Implementation*. John Wiley & Sons, Inc, 1994.

[2] J. Barraquand, J.-C. Latombe, "Robot Motion Planning: A Distributed Representation Approach," *Int. J. of Rob. Res.*, Vol.10, №6, pp.628-649, December 1991.

[3] V. A. Ilyin, *Intelligent Robots: Theory and Algorithms.* Krasnoyarsk. SAA, 1995 (in Russian).

[4] S. M. LaValle, *Planning Algorithms*, 1999-2004. Available: http://msl.cs.uiuc.edu/planning.

[5] C. S. G. Lee "Robot Arm Kinematics, Dynamics and Control," *CompSAC 82: Proc. IEEE Comput. Soc. 6-th Int. Comput. Software And Appl. Conf.*, Chicago, Ill., Nov.8-12, 1982 - pp.601-610.

[6] P. K. Lopatin, "Algorithm of a manipulator movement amidst unknown obstacles". *Proc. of the 10th International Conference on Advanced Robotics (ICAR 2001),* August 22-25, 2001, Hotel Mercure Buda, Budapest, Hungary. pp.327-331.

[7] P. K. Lopatin, "Algorithm2 for Dynamic Systems' Control in an Unknown Static Environment". *Herald of The Siberian state aerospace university named after academician M.F.Reshetnev / ed. prof. G.P.Belyakov; SibSAU.* № 4(11). Krasnoyarsk. pp.28-32, 2006. (in Russian).

[8] P. K. Lopatin, A. S. Yegorov, "Using the Forward Search and the Polynomial Approximation Algorithms for Manipulator's Control in an Unknown Environment", *Proceeding of the 2006 IEEE Conference on Automation Science and Engineering.* Shanghai, China, October 7-10, 2006. pp.216-221.

[9] V. J. Lumelsky "Sensing, Intelligence, Motion : How Robots and Humans Move in an Unstructured World", John Wiley & Sons, 2006.

[10] N. Nilson, *Problem-Solving Methods in Artificial Intelligence.* McGraw-Hill Book Company, New York, 1971.

[11] F. Yegenoglu, A. M. Erkmen, H.E. Stephanou, "On-line Path Planning Under Uncertainty," *Proc. 27th IEEE Conf. Decis. and Contr.*, Austin, Tex., Dec.7-9, 1988. Vol.2, pp.1075-1079, New York (N.Y.), 1988.