

number of other problems can be effectively solved only based on their synthetic joint consideration. For example, for the translation of the word-combination “up-link communication” not into the English language as “communication with a satellite” it is necessary to use a visual image of the facts discussed in the text. This way, in a system of translations, while the text translating is a semantically visual image of narration that should be grown, a translation without a latter close to the human one is impossible.

For the realization of the principles aforementioned, it is necessary to start the elaboration of the dictionary for semantic trees of heterogeneous data: images, patterns of sentences composition, algorithms, and so on. It will be necessary to use the already existing dictionaries of sentences generation in the “Electronic Dictionary” software for the system basis.

In conclusion it is necessary to mention that generative grammar over the trees of strings is an effective means of generating the state trees for such systems, like natural language sentence and semantically loaded images. It is thought to apply the generative grammars over the trees of strings on the basis of the Semantic Trees’ Dictionary, which is a classification of heterogeneous semantic data.

References

1. Agamdjanova V. I. Contextual Redundancy of the Lexical Meaning of a Word. M. : Higher School, 1977.
2. Apresyan Yu. D. Ideas and Methods of Modern Structural Linguistics. M. : Science, 1966.
3. Verdieva Z. N. Semantic Fields in the Modern English Language. M. : Higher School, 1986.
4. Lichargin D. V. Operations over the Natural Language Words Semes in Machine Translation // Works of the Conf. of Young Scientists. Krasnoyarsk : ICM SB RAS, 2003. P. 23–31.
5. Lichargin D. V. Elimination of Semantic Noise as the Means of Adequate Translation // Questions of the Theory and Practice of Translation. Works of All-Russian Conference. Penza : Privolzhye Region Knowledge House, 2003. P. 90–92.
6. Lichargin D. V. Generation of the Natural Language Phrases within the Task of Creating Natural Language Interface with Software // Problems of the Territory Information Development : Materials of the Eighth All-Russian Conf. PTID. 2003. Vol. 2. Krasnoyarsk : IP Centre of Krasnoyarsk State Technical University, 2003. P. 152–156.
7. Nikitin M. V. Lexical Meaning of a Word. M. : Higher School, 1983.

© Lichargin D. V., 2010

P. K. Lopatin

Siberian State Aerospace University named after academician M. F. Reshetnev, Russia, Krasnoyarsk

AN ALGORITHM FOR AN OBJECT GRASPING BY A MANIPULATOR IN AN UNKNOWN STATIC ENVIRONMENT

An algorithm for a n-link manipulating robot (MR) control in an environment with unknown static obstacles is considered. A theorem is proved which states that following the algorithm a MR in a finite number of steps will either grasp an object or will give a proved conclusion that an object cannot be grasped in any configuration.

Keywords: robot, unknown environment, obstacles, reachability.

In MR control the following typical problem arises: a MR should move from a start configuration $\mathbf{q}^0$ and grasp an object *Obj* by its gripper. Herewith sometimes the *Obj* may be grasped not in one but in several and sometimes in an infinite number of target configurations $\mathbf{q}_i^T$. The target configurations are united into a target set B_T . The set B_T has an arbitrary shape.

Let us consider that the B_T does not grow during the whole movement of MR. Consider also that the coordinates of every point from B_T are known and defined reliably.

A MR is represented in the configuration space (generalized coordinate space) as a point. MR functioning should take place in the bounded region X of the configuration space. Let's consider that X is such that for any $\mathbf{q} \in X$ the following inequalities are fulfilled:

$$\mathbf{a}^1 \leq \mathbf{q} \leq \mathbf{a}^2, \quad (1)$$

where $\mathbf{a}^1 = (a_1^1, a_2^1, \dots, a_n^1)$ is a vector of lower bounds on the generalized coordinates values, $\mathbf{a}^2 = (a_1^2, a_2^2, \dots, a_n^2)$ is a vector of upper bounds on the generalized coordinates values of a MR, $\mathbf{q} = (q_1, q_2, \dots, q_n)$ is a vector of the generalized coordinates of a MR. So X is a hyper parallelepiped. We will consider all points not satisfying (1) as forbidden.

Moreover, it is necessary to take into account that there also may be forbidden states inside X . Firstly these are the states (configurations) conditioned by constructive limitations of a MR, for example, those in which inadmissible intersection of MR links takes place. It is possible to calculate such forbidden configurations in advance. Secondly we will consider a configuration as forbidden in case when it intersects obstacles. It is impossible to calculate all such configurations in advance in the conditions of an unknown environment. So we will consider a configuration as forbidden if a MR cannot be

present in it because of constructive limitations or because of intersection with an obstacle. Before the MR movement beginning we do not have information about forbidden states in X or it is incomplete. If we do not have exact information that a point $\mathbf{q}^* \in X$ is forbidden, we will consider such point as allowed.

Now let us consider points from B_T . We will consider a point $\mathbf{q}^T \in B_T$ as allowed if it satisfies both criteria: 1) it is not forbidden in senses described in previous paragraphs, 2) it may be reached from $\mathbf{q}^0$ in a finite number of steps moving in X by allowed states. We will consider points from B_T as forbidden if they do not satisfy at least one criterion. As far as we need to find out whether the set B_T is reachable at least in one point we will consider that before the MR movement we do not have information about any point from B_T whether it is forbidden or allowed. Now let us formulate the following Problem of a MR control in an unknown static environment: a start configuration $\mathbf{q}^0$ and a target set B_T are given. It is necessary to propose an algorithm which in a finite number of steps will either move the MR from $\mathbf{q}^0$ to a point from B_T or will give a proved conclusion that there is no allowed state in B_T .

Review of related works. Currently there are many works dedicated to algorithms for dynamic systems (DS) control and, in particular, for robotic systems (RS) in known and unknown environments. There are good reviews of such algorithms [1; 2]. There are algorithms, for example, the forward search algorithm and the A^* algorithm [3] which in a finite number of steps will either find a path from $\mathbf{q}^0$ to a point from B_T or inform that there is no point in B_T reachable from $\mathbf{q}^0$.

Some algorithms for planning in a known environment in principle may be used for movement in an unknown environment. If we discretize the configuration space then we may use graph algorithms for a DS path searching [2; 3]. But these algorithms have one common feature which makes their application for the DS control in an unknown environment very difficult. The feature is that they demand to carry out the breadth-first search in a certain volume otherwise reaching of a target point $\mathbf{q}_i^T$ is not guaranteed [4]. But during the breadth-first search the following situation often arises: suppose we have just finished considering the vertices adjacent to a vertex $\mathbf{q}$ and we have to consider vertices adjacent to a vertex $\mathbf{q}'$ and the $\mathbf{q}$ and $\mathbf{q}'$ are not adjacent. In order to consider vertices adjacent to the $\mathbf{q}'$ the manipulator has to come to the $\mathbf{q}'$ at first. So we get a problem of the manipulator movement from $\mathbf{q}$ to $\mathbf{q}'$. The necessity of searching and following paths for multiple different $\mathbf{q}$ and $\mathbf{q}'$ makes the total sum of the manipulator movements very big [4]. In case when we plan a path in a known environment a computer simply “switches its attention” from $\mathbf{q}$ to $\mathbf{q}'$, which are stored in its memory.

According to classification [2] it is possible to outline the following representatives of the breadth-first approach: proper breadth-first searching algorithm, A^* algorithm, best-first heuristic search, lazy PRM, dynamic

programming. The methods based on a randomized potential field, Ariadne’s Clew algorithm, rapidly-exploring random trees [2] have such feature that new vertices are generated randomly and therefore using these methods for the unknown environment leads to the same difficulties. The approaches based on cell decomposition, visibility (bitangent) graphs, Voronoi diagrams [2] are reduced to alternate graph building and searching a path on it and have the above mentioned disadvantage connected with multiple mechanical movements.

In the algorithm presented in this article the vertices $\mathbf{q}$ and $\mathbf{q}'$ are always neighbor vertices and it reduces the number of movements.

It is also known that the “depth-first” algorithms do not guarantee reaching the goal [4].

There is a common difficulty for the methods of path planning in the presence of known obstacles: it is very difficult to borrow full information about workspace of a robot in advance and to represent this information in a form suitable for trajectory planning. Considering our algorithm one can see that there is no need for a control system to have full information about workspace in advance, a manipulator will borrow necessary information by itself in limited quantities and in terms of generalized coordinates which is suitable for path planning.

The attempts of creating algorithms for the robot control in presence of unknown obstacles were made. Most of them cover various two-dimensional cases [5].

In [6–9] different approaches for a robot control in two-dimensional unknown environment are considered. In [6; 9] the approaches are based on Voronoi diagrams, in [8] a tabu search approach is presented. As these approaches demand alternate graph building and searching a path on it they lead to multiple robot movements. In [7] obstacles should have polygonal form. The application of methods proposed in [6–9] to a n -link manipulator control in an unknown environment is not presented.

In [5] an algorithm for the control of manipulators in the presence of unknown obstacles in three-dimensional space is given. A MR must have not more than three elements and the last kinematic pair should be sliding. In the given conditions the algorithm in a finite number of steps either transfers a MR into a target configuration or informs that it is unreachable.

In [10] the n -dimensional case is considered. The algorithm is based on the solution of the system of nonlinear equations using Newton method and therefore it cannot guarantee the reaching of a target position.

In [2] algorithms for moving a robot in the presence of uncertainty (including cases of an unknown environment) are considered. The algorithms are based on the sequential solution theory. In general 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.

In [11] an algorithm for controlling dynamic systems in an n -dimensional state space in presence of unknown forbidden static states is proposed. The algorithm in a finite number of steps either moves the DS from a start point q^0 to a target point q^T or gives a proved conclusion that the q^T is unreachable. The algorithm's disadvantage is that it assumes that in the set X any state may be reached from any state. In a number of cases such demand may contradict the purpose of the algorithm to define in a finite number of steps whether a target state is reachable from a start state.

In [4] an approach to robot's (including manipulating robots) control in a n -dimensional state space is proposed. The essence of this approach is that the robot generates a path, connecting a start point q^0 and a target point q^T , not intersecting known forbidden states and tries to follow the trajectory either till reaching the q^T or till meeting an earlier unknown forbidden state in a point q^n . In the last case a new path $L(q^n, q^T)$ is generated, connecting q^n and q^T and not intersecting any known forbidden state. It is shown [4] that the problem of the robot movement from q^0 to q^T in an unknown environment is reduced to a solution of a finite number of problems of generating and following the path $L(q^n, q^T)$ (in other words to a solution of a finite number of problems PI (planning in a known environment)). Herewith it is supposed that we have a priori information that the q^T is reachable.

In this article we used the approach [4] for a solution of the Problem. The Problem is reduced to investigation of reachability of a finite number N_{BT} of points q_i^T , $I = 1, 2, \dots, N_{BT}$. The demand of the a priori knowledge of q_i^T , $I = 1, 2, \dots, N_{BT}$ reachability is omitted. New demands to the PI procedure necessary for the Problem solution were formulated.

In [12–15] algorithms were given which in a finite number of steps move a DS from q^0 to q^T in an environment with unknown static states. Herewith it was supposed that there is a priori information that q^T is reachable.

Preliminary conditions

1. Let us extract from the set B_T a finite number N_{BT} of points q_i^T , $I = 1, 2, \dots, N_{BT}$. These are the configurations whose reachability will be explored. Further we will consider B_T as a list of configurations q_i^T , $I = 1, 2, \dots, N_{BT}$. Let's consider that B_T is not replenished and therefore N_{BT} is not increased, consider that coordinates of every point from B_T are defined reliably.

2. Consider that we have a procedure PI which in a finite number of steps either generates a path from an arbitrary allowed point $q^n \in X$ to an arbitrary point $q^T \in X$ in the presence of known forbidden states or informs that q^T is unreachable. Such procedures already exist, for example, the forward search algorithm or the A^* algorithm [3], which for any start point q^n and any target point q^T under given known forbidden states in a finite number of states either find a path from q^n to q^T or inform that a path from q^n to q^T cannot be found.

3. The obstacles' disposition inside the MR working area does not change during the whole time of the MR movement.

4. The obstacles' number inside the MR working area does not change during the whole time of the MR movement.

5. The MR movement including the resultant path should take place inside the hyperparallelepiped (1).

6. The MR has a sensor system (SS) which supplies information about a r -neighborhood of a current MR point $q^n \in X$. The current point of the MR is the point where the MR is situated right now. The r -neighborhood of the q is a hyperball in X with a center in q and radius $r > 0$. We denote the set of all points comprising the r -neighborhood of the q as $Y(q)$. The words "supplies information about the r -neighborhood of the point q " mean that the SS defines whether every point from $Y(q)$ is allowed or forbidden. Herewith all forbidden points from $Y(q)$ are stored in a set $Q(q)$ and all allowed points from $Y(q)$ are stored in a set $Z(q)$. There may be different ways of the sets $Y(q)$, $Q(q)$, $Z(q)$ representation – in a form of formulas, lists, tables etc., but we consider that we have such representation. We will not consider the SS structure.

7. Consider that we have a program *Procedure1*($B_T, N_{BT}, Q(q^n)$). *Procedure1*($\cdot$) in the moment of call gets the set B_T , the number N_{BT} of points in the set B_T , the set $Q(q^n)$ which was formed during the last call of the SS. *Procedure1*($\cdot$) throws out of the B_T those points which coincide with points from $Q(q^n)$. After the throw the points left in the B_T are reenumerated by the continuous numeration beginning from 1 and the number of points left in B_T after execution of *Procedure1*($\cdot$) is inscribed in N_{BT} .

8. Consider that we have a program *Procedure2*(B_T, N_{BT}, q^T). *Procedure2*($\cdot$) in the moment of call gets the set B_T , the number N_{BT} of points in the set B_T and the point q^T . *Procedure2*($\cdot$) throws out of the B_T the point q^T . After that the points are reenumerated by the continuous numeration beginning from 1 and the number of points left in B_T after execution of *Procedure2*($\cdot$) is inscribed in N_{BT} .

Below the Algorithm for the solution of the Problem is given. Before a movement beginning the current configuration q^c of the MR is q^0 , during the movement the Algorithm 1 may be called from other current configurations of the MR.

Algorithm

If $N_{BT} = 0$ then the Algorithm terminates its work with a message that the *Obj* cannot be grasped. Otherwise we consider that the first point from B_T is q^T .

STEP 1. The MR is in a configuration q^c . $n = 0$, $q^n = q^c$. Execute target_point_is_forbidden: = Algorithm 1 (q^c, q^T, B_T, N_{BT}). If target_point_is_forbidden: = *NO* then the Algorithm successfully terminates its work with a message that the object is grasped in the point q^T .

If target_point_is_forbidden = *YES* go to STEP 2.

STEP 2. If $N_{BT} = 0$ the Algorithm terminates its work with a message that an object cannot be grasped in any target configuration. If $N_{BT} \neq 0$ consider the first point from B_T as q^T and go to STEP 1. The End of the Algorithm.

Algorithm 1 gets values in the format: Algorithm 1(q^n , q^T , B_T , N_{BT}) and defines whether point q^T is reachable from q^n in an unknown environment or not.

Algorithm 1

STEP 1. MR is in q^n (let us call it “a path changing point”). SS supplies information about $Y(q^n)$, $Z(q^n)$, $Q(q^n)$.

STEP 2. Execute

$$N_{BT} := \text{Procedure1}(B_T, N_{BT}, Q(q^n)).$$

If $N_{BT} = 0$ then target_point_is_forbidden: = YES and return to the Algorithm. If $N_{BT} \neq 0$ check whether the point q^T was thrown out of B_T . If yes then target_point_is_forbidden: = YES and return to the Algorithm, if no (that is q^T was left in B_T) go to STEP 3.

STEP 3. Call the PI procedure in order to generate a path $L(q^n, q^T)$ satisfying the following conditions:

– $L(q^n, q^T)$ connects q^n and q^T ;

– $L(q^n, q^T)$ does not intersect the set $\bigcup_{i=0}^n Q(q^n)$ that is, it

does not intersect any forbidden point;

– $L(q^n, q^T)$ satisfies the limitations (1).

There may be two results of the PI procedure execution:

1. PI returns the generated path and Algorithm 1 goes to STEP 4;

2. PI informs that the $L(q^n, q^T)$ cannot be generated, that is, the q^T is unreachable. In this case:

$$N_{BT} := \text{Procedure2}(B_T, N_{BT}, q^T),$$

make assignment target_configuration_is_forbidden: = YES and return to the Algorithm.

STEP 4. MR begins to follow the $L(q^n, q^T)$. There may be two results:

1. MR comes to a point $q_i^T \in B_T$. In this case make assignments $q^T := q_i^T$ and target_point_is_forbidden: = NO and return to the Algorithm;

2. MR will come to such point q^* that the next point after that is forbidden. In this case execute $n := n + 1$, $q^n := q^*$ and Algorithm1 goes to STEP 1. The End of Algorithm 1.

Theorem. Executing the Algorithm the MR will solve the Problem in a finite number of steps.

Proof. The Algorithm defines the reachability of a finite number of points from B_T . In order to define whether a point $q_i^T \in B_T$ is reachable it is necessary to call Algorithm 1 one time. Therefore the Algorithm execution is reduced to a finite number of calls of Algorithm 1. Therefore in order to demonstrate that the Algorithm will be executed in a finite number of steps it is necessary to show that Algorithm 1 will be executed in a finite number of steps for arbitrary q^n and q^T .

Algorithm 1 defines whether a point q^T is reachable in an unknown environment from a path changing point q^n or not. In Algorithm 1 when the MR is in point q^n , $n = 0, 1, 2, \dots$ the SS and the PI procedure are called. If after executing of these actions the point q^T will be defined as forbidden (because of intersection with obstacles or unreachability) the return to the Algorithm will take place and another point $q_i^T \in B_T$ will be considered. If after execution of these actions the point q^T will not be defined as forbidden, a path $L(q^n, q^T)$ will be generated and the MR will begin to follow this path. There may be two results of following this path: either MR will not meet forbidden points and therefore will reach the q^T (and therefore successful termination of the Algorithm work will occur) or the MR will come to such a point q^n , $n = 1, 2, \dots$ that the next point will be forbidden. Let us show that all path changing points q^n , $n = 0, 1, 2, \dots$ will be different and their number will be finite.

Let us prove that all points where the MR changes its trajectory will be different. Suppose that the manipulator changed its trajectory being in point q^s , and later it again changed its trajectory, being in point q^p , that is $s < p$. Let us show that $q^s \neq q^p$. Suppose, at first, that, on the contrary $q^s = q^p$. Then $Q(q^s) = Q(q^p)$. As the manipulator changed its trajectory being in point q^s it generated the trajectory which did not intersect the sets $Q(q_i)$, $i = 0, 1, \dots, s$. But as it changed the trajectory in point q^p it means that its trajectory intersected $Q(q^p) = Q(q^s)$ (besides, $q^s = q^p$ is the center of r -neighborhood of the point $q^s = q^p$ and the following point is forbidden). That is, $Q(q^p) = Q(q^s)$ was unknown. Here is a contradiction. It means that all points where the manipulator changes its trajectory are different.

Now let us show that the number of such points is finite. Suppose that it is infinite. All points of a trajectory changing must satisfy the inequalities (1). It means, that the sequence of these points is limited. According to the Boltsano-Weierstrass theorem it is possible to extract a convergent subsequence q_i , $i = 1, 2, \dots$ from this sequence According to Cauchy property of the convergent sequences it is possible for any ε to find such a number s that all points q_i , $i > s$ will lie in an ε -neighborhood of q_s . Let us take $\varepsilon < r$. Consider an arbitrary point q_i of the trajectory changing lying in the ε -neighborhood of q_s . As the manipulator had to change the trajectory in the q_i , it means that that trajectory intersected $Q(q_s)$ (because q_i and its neighbor points belong to $Q(q_s)$). From this fact it is possible to draw the conclusion that the set $Q(q_s)$ was not taken into account when that trajectory was generated. But such situation is impossible if we strictly follow the conditions of the Algorithm 1. The situation when a trajectory changing point belongs to the ε -neighborhood of another trajectory changing point will necessarily appear if the number of the points where trajectory is changing is infinite. But we showed that such situation is impossible and it means that a number of the points where it is necessary to change trajectory will be finite.

So, the number of path changing points q^n , $n = 0, 1, 2, \dots$ is finite and they are different. In every point q^n the SS and the procedure PI are called and PI generates a $L(q^n, q^T)$. As a result we either get information that q^T is forbidden or do not get it. If we get the information that q^T is forbidden then we consider the q^T as unreachable. Otherwise an attempt of the path $L(q^n, q^T)$ following occurs. If in the last path changing point q^n the point q^T was not qualified as forbidden, a path $L(q^n, q^T)$ will be generated, this path will be followed and the q^T will be reached.

So it was shown that executing Algorithm 1 the MR will either reach the q^T or will make the conclusion that the q^T is unreachable. The Algorithm is reduced to a finite number of the Algorithm 1 calls. Therefore one may see that executing the Algorithm the MR will solve the Problem in a finite number of steps. The Theorem is proved.

Note 1. We have already mentioned that Algorithm 1 is reduced to a finite number of paths $L(q^n, q^T)$ generation and following. The Algorithm is reduced to a finite number of the Algorithm 1 calls. Therefore one may see that the Problem is reduced to the solution of a finite number of PI problems of a path generating and following in an environment with known forbidden states.

Note 2. On the first call of the Algorithm 1 $q^c = q^0$, on the next calls, generally speaking, $q^c \neq q^0$. Upon the Algorithm 1 execution a conclusion about a q^T reachability/unreachability from a q^c is drawn. But, as the MR has come to q^c from q^0 by continuously following each other allowed points, the conclusion about reachability/unreachability of q^T from q^c will be simultaneously considered as the conclusion about reachability/unreachability of q^T from q^0 .

An algorithm for a n-link manipulating robot control in an environment with unknown static obstacles was considered. A theorem was proved which states that following the algorithm the MR will either grasp an object or will give a proved conclusion that an object cannot be grasped in any configuration in a finite number of steps.

References

1. Ahrikhencheikh C., Seireg A. Optimized-Motion Planning: Theory And Implementation. John Wiley & Sons, Inc, 1994.
2. LaValle S. M. Planning Algorithms. 1999–2006. URL: <http://msl.cs.uiuc.edu/planning>.
3. Nilson N. Problem-Solving Methods in Artificial Intelligence. N. Y. : McGraw-Hill Book Company, 1971.
4. Ilyin V. A. Intellectual robots: theory and algorithms. Krasnoyarsk : Publishing house SAA, 1995.
5. Lumelsky V. J. Sensing, intelligence, motion. How robots and humans move in an unstructured world. John Wiley & Sons, 2006.
6. Chen C., Li H.-X., Dong D. Hybrid Control for Robot Navigation. A hierarchical Q-learning algorithm // IEEE Robotics & Automation Magazine. 2008. Vol. 15, № 2. P. 37–47.
7. Online Algorithms with Discrete Visibility. Exploring Unknown Polygonal Environments / S. K. Ghosh, J. W. Burdick, A. Bhattacharya, S. Sarkar // IEEE Robotics & Automation Magazine. 2008. Vol. 15, № 2. P. 67–76.
8. Masehian E., Amin-Nasari M. R. Sensor-Based Robot Motion Planning. A Tabu Search Approach // IEEE Robotics & Automation Magazine. 2008. Vol. 15, № 2. P. 48–57.
9. Rawlinson D., Jarvis R. Ways to Tell Robots Where to Go. Directing autonomous robots using topological instructions // IEEE Robotics & Automation Magazine. 2008. Vol. 15, № 2. P. 27–36.
10. Yegenoglu F., Erkmen A. M., Stephanou H. E. On-line Path Planning Under Uncertainty // Proc. 27th IEEE Conf. Decis. and Contr. (Austin, Tex. Dec.7–9, N. Y.). 1988. Vol. 2. P. 1075–1079.
11. Algorithmic provision and software of intellectual manipulation robots in conditions of uncertainty. Theory and modelling of control tasks, trajectories planning and actions of manipulation intellectual robots in conditions of uncertainty : report on state budget scientific-research work B6-12 for 1996 / Thesis supervisor V. A. Ilyin. Krasnoyarsk, 1997. № state registration 01.9.80.001231. Inv. № 02.9.8.0000526.
12. Lopatin P. K. Computer simulation of manipulation robots control in unknown environment on the basis of exact and simplified algorithms // Mechatronics, automatization, control. Appendix. 2006. № 8. P. 7–14.
13. Lopatin P. K. Algorithm of dynamic systems control in unknown static environment // Vestnik. Scientific Journal of Siberian aerospace university named after academician M. F. Reshetnev. Issue. 4(11). 2006. P. 28–32.
14. Lopatin P. K. Algorithm of dynamic systems control in unknown static environment // Mechatronics, automatization, control. № 2. 2007. P. 9–13.
15. Lopatin P. K. Algorithm of a manipulator movement amidst unknown obstacles // Proc. of the 10th Intern. Conf. on Advanced Robotics (August 22–25, Budapest). Budapest : Hotel Mercure Buda, 2001. P. 327–331.