ISSN 1992-6502 (Print)
2014. Т. 18, № 1 (62). С. 198–203
Вестник УГАТУ
ISSN 2225-2789 (Online)
http://journal.ugatu.ac.ru

УДК 519.713

Модифицированный алгоритм полного перебора в задаче управления манипуляционным роботом в неизвестной среде

П. К. Лопатин

efa14@yandex.ru

ФГБОУ ВПО «Сибирский государственный аэрокосмический университет им. акад. М. Ф. Решетнева» (СибГАУ)

Поступила в редакцию 22.06.2013

Аннотация. Рассматривается алгоритм управления *n*-звенным манипуляционным роботом (МР) в среде с неизвестными статическими препятствиями. Алгоритм сводится к решению конечного числа задач ПИ планирования пути в среде с уже обнаруженными, а потому известными запрещенными состояниями с последующим исполнением пути. Приведен алгоритм решения задачи ПИ. Доказана теорема, утверждающая, что этот алгоритм решит задачу ПИ за конечное число шагов.

Ключевые слова: робот; неизвестная среда; препятствия; достижимость.

При управлении манипуляционным роботом (МР) типичной является следующая задача: МР должен выдвинуться из стартовой конфигурации $\boldsymbol{q}^0$ и захватить своим схватом некоторый объект Obj. При этом часто Obj может быть захвачен не в одной, а во многих целевых конфигурациях $\boldsymbol{q}_i^T$. Целевые конфигурации $\boldsymbol{q}_i^T$ объединяются в целевое множество B_T. Множество B_T имеет произвольный вид. Будем считать, что B_T не пополняется в течение всего времени движения МР. Считаем также, что координаты каждой точки из B_T известны и определены достоверно.

МР представляется в пространстве конфигураций (пространстве обобщенных координат) как точка. Функционирование МР должно происходить в пределах ограниченной области X конфигурационного пространства. Будем считать, что область X имеет такой вид, что для любого $\boldsymbol{q} \in X$ выполняются неравенства:

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

где $\mathbf{a}^1 = (a_1^1, a_2^1, \ldots, a_n^1)$ – вектор нижних ограничений на значения обобщенных координат, $\mathbf{a}^2=(a_1^2, a_2^2, \ldots a_n^2)$ – вектор верхних ограничений на значения обобщенных координат, $\boldsymbol{q} = (q_1, q_2, \ldots, q_n)$ – вектор обобщенных координат МР. Таким образом, область X представляет собой гиперпараллелепипед. При этом:

1) точки, находящиеся вне (1), квалифицируются как запрещенные;

2) внутри X также могут присутствовать запрещенные состояния, которые можно вычислить заранее, например, это те состояния (конфигурации), в которых происходит недопустимое взаимопересечение звеньев.

Кроме того, запрещенной является та конфигурация, в которой МР налегает на препятствия. В условиях неизвестной среды все такие конфигурации вычислить заранее невозможно. Перед началом движения информации о запрещенных состояниях в X нет или она неполна. Точки из X, про которые нет достоверной информации о том, что они запрещенные, считаем разрешенными.

Рассмотрим теперь точки из B_T. Достижимой точкой $\boldsymbol{q}^T \in B_T$ будем считать только ту точку, которая удовлетворяет следующим двум критериям: 1) она не является запрещенной, 2) в нее можно попасть за конечное число шагов из $\boldsymbol{q}^0$, двигаясь в X по разрешенным состояниям. Точки из B_T, не удовлетворяющие хотя бы одному из двух таких критериев, считаем недостижимыми. Теперь сформулируем следующую ***Задачу*** управления МР в неизвестной статической среде: даны стартовая конфигурация МР $\boldsymbol{q}^0$ и целевое множество B_T. Требуется предложить алгоритм, который за конечное число шагов либо передвинет МР из $\boldsymbol{q}^0$ в хотя бы одно достижимое состояние из множества B_T, либо выдаст обоснованный ответ о том, что ни одно состояние из целевого множества B_T не является достижимым.

ВВОДНЫЕ ЗАМЕЧАНИЯ

Если предположить, что B_T состоит из конечного числа точек $(\boldsymbol{q}_1^T, \boldsymbol{q}_2^T, \ldots \boldsymbol{q}_N^T)$, то для решения ***Задачи*** можно применять различные алгоритмы управления МР в неизвестной и известной средах. Большинство исследователей предлагают использовать графовые алгоритмы. Однако эти алгоритмы имеют одно общее свойство, которое затрудняет их применение для неизвестной среды, заключающееся в том, что они в том или ином объеме требуют осуществлять поиск в ширину, иначе не гарантируется достижение цели [1]. Но при поиске в ширину часто возникает следующая ситуация: предположим, что мы только что закончили рассмотрение вершин, соседних к вершине $\boldsymbol{q}$, и теперь нам надо рассматривать вершины, соседние вершине $\boldsymbol{q}'$, и вершины $\boldsymbol{q}$ и $\boldsymbol{q}'$, не являются соседними. Для того чтобы рассмотреть вершины, соседние к $\boldsymbol{q}'$, МР должен сначала физически передвинуться в $\boldsymbol{q}'$. Вследствие этого общая сумма передвижений МР становится очень большой.

В соответствии с классификацией [2], одним из представителей алгоритмов поиска в ширину является алгоритм полного перебора. Но в нашем случае он будет использоваться не для неизвестной среды, а как подпрограмма планирования пути в среде с известными запрещенными состояниями. В случае, когда мы планируем путь в известной среде, компьютер просто переключает свое «внимание» от точки $\boldsymbol{q}$ к точке $\boldsymbol{q}'$, которые хранятся в памяти компьютера. Данное обстоятельство открывает перспективы к снижению числа механических перемещений. Подробный обзор подходов к управлению МР в неизвестной среде проведен в [3].

ПРЕДВАРИТЕЛЬНЫЕ УСЛОВИЯ

1. На множестве X вводим дискретизацию, то есть накладываем на X сетку, и впредь будем рассматривать только точки, лежащие на узлах сетки. В силу введенной дискретизации число точек в X является конечным.

2. В силу введенной дискретизации число N_{BT} точек $\boldsymbol{q}_i^T$, образующих множество B_T, будет конечным. В дальнейшем B_T будем рассматривать как список конфигураций $\boldsymbol{q}_i^T$, i=1,…, N_{BT}. Считаем, что B_T не пополняется и потому N_{BT} не увеличивается. Считаем, что координаты каждой точки из B_T определяются достоверно.

3. Считаем, что у нас есть программная процедура ПИ ($\boldsymbol{q}^n$, $\boldsymbol{q}^T$, *ZAPR*, X), которая решает задачу ПИ (планирования пути в среде с известными запрещенными состояниями). Задачу ПИ сформулируем следующим образом: за конечное число шагов либо найти путь из $\boldsymbol{q}^n$ в $\boldsymbol{q}^T$, ни одной своей точкой не совпадающий ни с одной точкой из множества *ZAPR* и целиком лежащий в X – в случае, если хотя бы один такой путь существует, либо, если ни одного такого пути нет, обнаружить это. Алгоритм для процедуры ПИ() сформулирован в настоящей статье. Поскольку ПИ() будет вызываться в момент, когда МР будет находиться в $\boldsymbol{q}^n$, n=0,1,2,…, то $\boldsymbol{q}^n$ всегда будет разрешенной. Поскольку на момент вызова ПИ() еще нет информации – является $\boldsymbol{q}^T$ запрещенной или нет, то при вызове ПИ() $\boldsymbol{q}^T$ считается разрешенной. *ZAPR* – это множество (список) всех обнаруженных на момент вызова ПИ() запрещенных точек, поэтому их координаты известны. X – гиперпараллелепипед, определенный (1). Путь – это последовательность точек $\{\boldsymbol{q}^1=\boldsymbol{q}^n, \boldsymbol{q}^2, \boldsymbol{q}^3, \ldots, \boldsymbol{q}^m=\boldsymbol{q}^T\}$, исходящая из $\boldsymbol{q}^n$ и приводящая в $\boldsymbol{q}^T$. Для каждой точки $\boldsymbol{q}^j$, j = 2, 3, …, m пути выполняется правило: предшествующая ей точка пути $\boldsymbol{q}^{j-1}$, j = 1, 2, …, m–1 отстоит от $\boldsymbol{q}^j$ не более, чем на один дискрет (см. рис. 1). В случае, если процедуре ПИ() удается спланировать путь, она возвращает 1, в противном случае – 0.

4. Расположение и количество препятствий внутри рабочей зоны МР остается неизменным в течение всего времени движения МР.

5. Все движение, в том числе и результирующий путь, должно происходить в гиперпараллелепипеде (1).

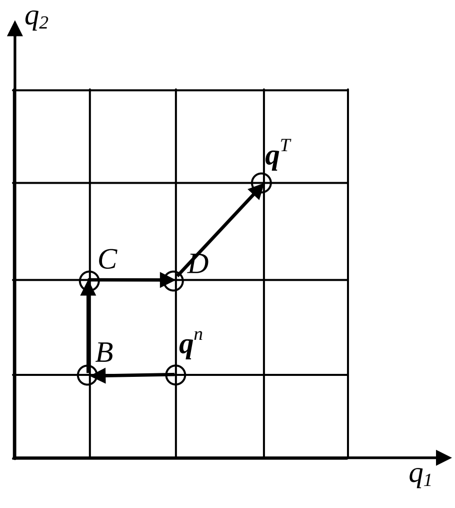

Рис. 1. Каждая последующая точка в пути $\boldsymbol{q}^n$-B-C-D-$\boldsymbol{q}^T$ отстоит от предыдущей не более чем на один дискрет

6. МР имеет сенсорную систему (СС), которая доставляет информацию об r-окрестности текущей точки МР $\boldsymbol{q}\in X$. Текущая точка МР – это та точка, в которой МР в настоящий момент находится.

Под r-окрестностью $\boldsymbol{q}$ понимаем множество точек, соседних к $\boldsymbol{q}$ и отстоящих от $\boldsymbol{q}$ не более, чем на один дискрет. Рис. 2 иллюстрирует расположение соседних точек (*A, B, C, D, E, F, G, H*) к точке $\boldsymbol{q}$ в двухмерном конфигурационном пространстве.

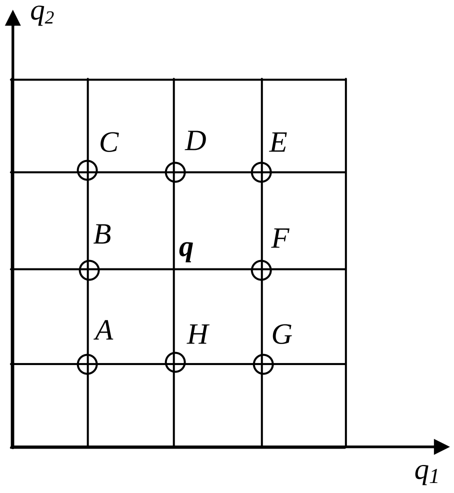

Рис. 2. Точки *A–H* образуют *r*-окрестность точки $\boldsymbol{q}$ в 2-мерном дискретизированном конфигурационном пространстве

Число соседних точек равно $3^n - 1$, где n – размерность конфигурационного пространства.

Множество всех точек, входящих в *r*-окрестность точки $\boldsymbol{q}$**,** обозначаем $Y(\boldsymbol{q})$ (в двухмерном конфигурационном пространстве $Y(\boldsymbol{q})$ будет состоять из 8 точек *A–H*). Слова «доставляет информацию об *r*-окрестности точки $\boldsymbol{q}$» означают, что относительно каждой точки из $Y(\boldsymbol{q})$ ее СС определяет, является ли она разрешенной или запрещенной, при этом все запрещенные точки из $Y(\boldsymbol{q})$ заносятся в множество $Q(\boldsymbol{q})$, а все разрешенные точки из $Y(\boldsymbol{q})$ заносятся в множество $Z(\boldsymbol{q})$.

Устройство СС в данной работе не рассматривается.

7. Считаем, что у нас есть программная *Процедура1*(B_T, N_{BT}, *ZAPR*). *Процедура1*(B_T, N_{BT}, *ZAPR*) получает при вызове множество B_T, количество N_{BT} точек в множестве B_T, множество *ZAPR* запрещенных точек. *Процедура1*(B_T, N_{BT}, *ZAPR*) выбрасывает из B_T те точки, которые совпадают с точками из *ZAPR*. После выброса оставшиеся точки в B_T перенумеровываются сплошной нумерацией начиная с 1 и в N_{BT} записывается число точек, оставшихся в B_T после исполнения *Процедуры1*(B_T, N_{BT}, *ZAPR*).

8. Считаем, что у нас есть программная *Процедура2*($\boldsymbol{q}$). *Процедура2*($\boldsymbol{q}$) работает в соответствии со следующим псевдокодом:

Процедура2($\boldsymbol{q}$)	
1	СС доставляет информацию о $Y(\boldsymbol{q})$, $Z(\boldsymbol{q})$, $Q(\boldsymbol{q})$.
2	*if* ($QBT{:}=Q(\boldsymbol{q})\cap B_T\neq 0$) /*то есть если поступила информация о том, что какие-то конфигурации из B_T являются запрещенными, произвести выброс запрещенных точек из B_T*/ N_{BT}:=*Процедура1*(B_T, N_{BT}, $Q(\boldsymbol{q})$); *endif*.
3	*return*(N_{BT});
4	*Конец Процедуры2*()

9. Исполнение пути происходит следующим образом. МР находится в текущей точке $\boldsymbol{q}$ и, если следующая точка $\boldsymbol{q}^*$ пути не является запрещенной, то МР переходит в $\boldsymbol{q}^*$. В противном случае в $\boldsymbol{q}^*$ МР не переходит.

АЛГОРИТМ ДЛЯ НЕИЗВЕСТНОЙ СРЕДЫ

Перед началом движения МР находится в $\boldsymbol{q}^0$. $\boldsymbol{q}^c$ – это текущая конфигурация МР, то есть конфигурация, в которой МР в настоящий момент находится. n – номер точки смены пути $\boldsymbol{q}^n$, n=0, 1, 2, …. Перед началом движения n=0, $\boldsymbol{q}^n = \boldsymbol{q}^c = \boldsymbol{q}^0$.

Алгоритм	
1	*n*:=0; *while* ($N_{BT}\neq 0$)
2	В качестве $\boldsymbol{q}^T$ выбираем первую точку из B_T
3	*if* (*объект_захвачен*:= ***Алгоритм1***($\boldsymbol{q}^c$, $\boldsymbol{q}^T$) =*ДА*) сообщить о том, что *Obj* захвачен в конфигурации $\boldsymbol{q}^T$; *go to* 6; *endif*
4	*endwhile*
5	Сообщить о том, что *Obj* не может быть захвачен
6	***Конец Алгоритма***

Алгоритм1 получает значения в формате: ***Алгоритм1***($\boldsymbol{q}^c$, $\boldsymbol{q}^T$) и посвящен выяснению вопроса о том, является ли точка $\boldsymbol{q}^T$ достижимой из $\boldsymbol{q}^c$ в неизвестной среде.

Алгоритм1($\boldsymbol{q}^c$, $\boldsymbol{q}^T$)	
1	$\boldsymbol{q}^n := \boldsymbol{q}^c$.
2	/*Запускаем СС и модифицируем B_T*/ N_{BT}:=*Процедура2*($\boldsymbol{q}^n$);
3	/*если в $Z(\boldsymbol{q}^n)$ есть конфигурации, в которых может быть захвачен *Obj**/ *if* ($ZBT{:}=Z(\boldsymbol{q}^n)\cap B_T\neq 0$) перейти в первую такую конфигурацию из B_T;

	объект захвачен:=ДА; *return* (объект_захвачен); *endif*
4	/*Если $\boldsymbol{q}^T$ оказалась выброшенной из B_T, то в качестве $\boldsymbol{q}^T$ надо назначать другую точку*/ *if* $(\boldsymbol{q}^T \notin B_T)$ объект_захвачен:=НЕТ; *return* (объект_захвачен); *endif*
5	/* Здесь осуществляется попытка планирования $L(\boldsymbol{q}^n, \boldsymbol{q}^T)$ внутри X*/ *if* $(ПИ\ (\boldsymbol{q}^n, \boldsymbol{q}^T, \bigcup_{i=0}^{n} Q(\boldsymbol{q}^i), X)=0)$ /*Если попытка не успешна,*/ /*$L(\boldsymbol{q}^n, \boldsymbol{q}^T)$ сгенерирован быть не может, то есть $\boldsymbol{q}^T$ является недостижимой*/ N_{BT}:=*Процедура1*$(B_T, N_{BT}, \boldsymbol{q}^T)$; объект_захвачен:=НЕТ; *return*(*объект_захвачен*); *endif* /*Если попытка успешна, происходит переход на 6*/
6	МР начинает исполнение пути $L(\boldsymbol{q}^n, \boldsymbol{q}^T)$. Исходов движения может быть два: 1) МР попадает в некоторую разрешенную точку $\boldsymbol{q}_i^T \in B_T$. В этом случае происходит присвоение $\boldsymbol{q}^T$:=$\boldsymbol{q}_i^T$ и *объект_захвачен:=ДА* и выполняется возврат в ***Алгоритм***; 2) МР придет в некоторую точку $\boldsymbol{q}^*$, следующая за которой является запрещенной. В этом случае n:=n+1; $\boldsymbol{q}^n$:=$\boldsymbol{q}^*$; *go to* 1;
7	***Конец Алгоритма1***

В работе [4] показано, что, исполняя ***Алгоритм***, МР решит ***Задачу*** за конечное число шагов. Это произойдет вследствие того, что число точек смены пути $\boldsymbol{q}^n$, n = 0, 1, 2, … будет конечным и при этом все они будут различны. В каждой точке $\boldsymbol{q}^n$ происходит пополнение информации от СС. В случае, если обнаруживается, что $\boldsymbol{q}^T$ недостижима, происходит возврат в ***Алгоритм***, в случае, если такой информации не поступило, предпринимается попытка решить задачу ПИ. В связи с этим возникает необходимость разработки алгоритма для решения задачи ПИ.

В [5] приведен алгоритм полного перебора, который за конечное число шагов планирует путь из стартовой точки в целевую. Однако этот алгоритм не учитывает возможности присутствия запрещенных состояний – таких, которые описаны в настоящей статье. Кроме того, не показано, что алгоритм выдает обоснованное утверждение о том, что путь не может быть найден тогда, когда такой случай имеет место.

Ниже мы приводим сформулированный нами модифицированный алгоритм полного перебора (МАПП). Сформулирована и доказана теорема, утверждающая, что МАПП решит задачу ПИ за конечное число шагов.

МОДИФИЦИРОВАННЫЙ АЛГОРИТМ ПОЛНОГО ПЕРЕБОРА ДЛЯ РЕШЕНИЯ ЗАДАЧИ ПИ

ПИ$(\boldsymbol{q}^n, \boldsymbol{q}^T, ZAPR, X)$	
1	Поместить $\boldsymbol{q}^n$ в начало списка ОТКРЫТ
2	*if* (ОТКРЫТ пуст) /*путь не может быть спланирован*/ *return*(0); *endif*
3	Взять первую вершину из начала списка ОТКРЫТ и поместить ее в ЗАКРЫТ. Обозначить эту вершину *curr*.
4	Раскрыть *curr*, то есть получить соседние к *curr* разрешенные вершины. Вершина считается соседней к *curr*, если она отстоит от *curr* не более чем на один дискрет. Вершина считается разрешенной, если она удовлетворяет всем следующим условиям: она удовлетворяет ограничениям (1), то есть лежит в X; она не налегает на препятствия, то есть ее нет в $ZAPR$; ее еще нет ни в списке ОТКРЫТ, ни в списке ЗАКРЫТ. Поместить полученные соседние к *curr* разрешенные вершины в конец списка ОТКРЫТ. Провести от каждой из них указатель к *curr* как к породившей их вершине.

5	*if* (какая-либо из полученных соседних разрешенных вершин совпадает с $\boldsymbol{q}^T$)
	/*путь спланирован*/
	return(1);
	else
	go to 2;
	endif

6	Конец ***ПИ()***

Теорема. Приведенный выше модифицированный алгоритм полного перебора решит задачу ПИ за конечное число шагов.

Доказательство. Согласно логике МАПП из списка ОТКРЫТ берется первая вершина, помещается в ЗАКРЫТ, для нее генерируются соседние вершины и те из них, которые удовлетворяют (1), не налегают на препятствия, а также не присутствуют в списках ОТКРЫТ или ЗАКРЫТ, помещаются в конец ОТКРЫТ. Таким образом, такой операции подвергается сначала вершина $\boldsymbol{q}^n$ (вершина 0-го уровня), для нее генерируются соседние разрешенные вершины (они являются вершинами 1-го уровня), $\boldsymbol{q}^n$ переносится в ЗАКРЫТ, вершины 1-го уровня оказываются в начале списка ОТКРЫТ, начинают раскрываться они, т. е. получаем вершины 2-го уровня и их помещаем в конец списка ОТКРЫТ, они оказываются за вершинами 1-го уровня. Отсюда видно, что приступить к раскрытию вершины 2-го уровня мы сможем только после того, как раскроем последнюю вершину 1-го уровня. Обобщая, делаем вывод, что приступить к раскрытию вершин i-го уровня мы сможем только после того как раскроем все вершины (i–1)-го уровня.

Покажем, что если хотя бы один путь от $\boldsymbol{q}^n$ к $\boldsymbol{q}^T$ существует, то он будет найден. Пусть $\boldsymbol{q}^T$ находится на n-м уровне. Поскольку путь от $\boldsymbol{q}^n$ к $\boldsymbol{q}^T$ существует, то это значит, что для $\boldsymbol{q}^n$ есть хотя бы одна соседняя разрешенная вершина 1-го уровня, для нее – хотя бы одна соседняя разрешенная вершина 2-го уровня и так далее, до n-го уровня (до $\boldsymbol{q}^T$), то есть имеется хотя бы одна цепочка из вершин, ведущих от $\boldsymbol{q}^n$ (вершины 0-го уровня) к $\boldsymbol{q}^T$ (вершине n-го уровня). Как было показано выше, МАПП просматривает все вершины i-го уровня (i = 0, 1, 2, …, n–1), генерируя все соседние к каждой из них разрешенные вершины (i+1)-го уровня, при этом уже имеются указатели от вершин i-го уровня к вершинам (i–1)-го уровня. При раскрытии каждой вершины i-го уровня будет проверяться, не совпадает ли получаемая соседняя вершина с $\boldsymbol{q}^T$. Таким образом, просматриваются все вершины каждого уровня и, если существует хотя бы одна цепочка вершин (путь) от $\boldsymbol{q}^n$ к $\boldsymbol{q}^T$, то она будет найдена.

Покажем теперь, что если пути от $\boldsymbol{q}^n$ к $\boldsymbol{q}^T$ не существует, то МАПП это обнаружит. Как было показано выше, МАПП помещает в ОТКРЫТ все разрешенные вершины каждого уровня, при этом к каждой вершине запоминается путь (цепочка вершин) из $\boldsymbol{q}^n$. Таким образом, хранятся все возможные пути из $\boldsymbol{q}^n$ к вершинам из ОТКРЫТ. Поступление сообщения «Список ОТКРЫТ пуст» означает, что из ОТКРЫТ был изъята последняя оставшаяся в нем вершина, и для нее, а также для всех остальных вершин из ОТКРЫТ были просмотрены соседние точки и среди них не оказалось $\boldsymbol{q}^T$. Таким образом, были исследованы все цепочки вершин из $\boldsymbol{q}^n$ и ни одна из них не привела в $\boldsymbol{q}^T$. Опустение списка ОТКРЫТ означает, что больше нет вершин, которые можно раскрыть. Таким образом, показано, что поступление сообщения «Список ОТКРЫТ пуст» является доказательством того, что пути $\boldsymbol{q}^n$ к $\boldsymbol{q}^T$ не существует.

Покажем, что задача ПИ будет решена за конечное число шагов. За все время работы МАПП в списке ОТКРЫТ побывает конечное число вершин, поскольку в силу введенной дискретизации во всем X будет конечное число вершин, при этом в ОТКРЫТ помещаются только разрешенные вершины. Каждая вершина помещается в ОТКРЫТ только один раз в силу требования 3) ШАГА 4 МАПП. В отношении каждой раскрываемой вершины, то есть изымаемой из списка ОТКРЫТ, проделываются следующие операции: 1) она помещается в список ЗАКРЫТ; 2) для нее генерируются соседние разрешенные вершины. Обе операции проделываются за конечное число шагов, вторая – в силу конечности числа соседних к раскрываемой вершин. Итак, из ОТКРЫТ будет изъято конечное число вершин и в отношении каждой из них будет проделано конечное число операций, каждая из которых состоит из конечного числа шагов. Если среди соседних к извлекаемой вершин обнаружится $\boldsymbol{q}^T$, путь найден, если и для последней извлеченной вершины не будет обнаружена $\boldsymbol{q}^T$, приходим к выводу, что путь найден быть не может. Отсюда видно, что задача ПИ будет решена за конечное число шагов. ***Теорема доказана***.

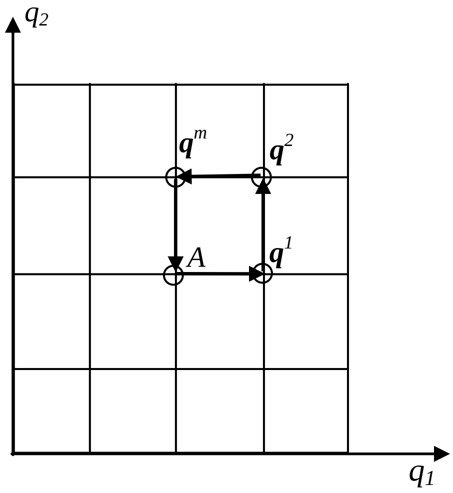

Рис. 3. Если вершину A вновь поместить в ОТКРЫТ, то путь, приведший в A будет получен вновь

Замечание. Покажем, что произойдет, если мы не будем соблюдать требование не помещать вновь какую бы то ни было вершину в ОТКРЫТ, если она уже есть либо в ОТКРЫТ либо в ЗАКРЫТ. Предположим, что мы раскрываем некоторую вершину, например, A. Для нее рассматриваются все соседние к ней вершины. Пусть, исполняя МАПП, мы получили цепочку вершин A–$\boldsymbol{q}^1$–$\boldsymbol{q}^2$–…–$\boldsymbol{q}^m$ (см. рис. 3). Среди соседних к вершине $\boldsymbol{q}^m$ имеется вершина A. Если мы вновь поместим А в ОТКРЫТ, то вновь придется раскрывать вершину A, то есть рассматривать все соседние к ней вершины и впоследствии вновь будет получен путь $A–\boldsymbol{q}^1–\boldsymbol{q}^2–…–\boldsymbol{q}^m$, то есть процесс повторится и повторяться он будет бесконечное число раз. Поэтому и было введено это требование. Заметим также, что если это требование будет соблюдаться, то каждая вершина, получаемая в качестве разрешенной соседней, будет иметь только одного родителя (непосредственно предшествующую ей вершину) и тогда граф, образующийся в результате исполнения МАПП, становится деревом.

ЗАКЛЮЧЕНИЕ

Рассмотрен алгоритм управления *n*-звенным манипуляционным роботом (МР) в среде с неизвестными статическими препятствиями. Алгоритм сводится к решению конечного числа раз задачи ПИ планирования пути в среде с известными запрещенными состояниями с последующим исполнением пути. Приведен алгоритм решения задачи ПИ, представляющий собой модифицированный алгоритм полного перебора. Доказана теорема, утверждающая, что этот алгоритм решит задачу ПИ за конечное число шагов.

СПИСОК ЛИТЕРАТУРЫ

1. **Ильин В. А.** Интеллектуальные роботы: Теория и алгоритмы. Красноярск: Изд-во САА, 1995. **[** V. A. Ilyin, *Intelligent robots: Theory and algorithms*, (in Russian). Krasnoyarsk: SAA Press, 1995. **]**
2. **LaValle S. M.** Planning Algorithms, 1999–2006. URL: http://msl.cs.uiuc.edu/planning (дата обращ.: 05.02.2013). **[** S. M. LaValle, (2013, Feb. 05), *Planning Algorithms, 1999–2006.* Available: http://msl.cs.uiuc.edu/planning **]**
3. **Лопатин П. К.,** Алгоритм перемещения манипуляционного робота в среде с неизвестными препятствиями // Вестник СибГАУ. 2011. Вып. 3 (36). С. 53–57. **[** P. K. Lopatin, “Algorithm of a manipulator movement in an environment with unknown obstacles,” (in Russian), *Vestnik SibSAU* (scientific journal of Siberian State Aerospace University), vol. 36, no. 3. pp.53-57, 2011. **]**
4. **Лопатин П. К.** Алгоритм исследования достижимости объекта манипулятором в неизвестной среде // Мехатроника, автоматизация, управление. 2012. № 9. С. 49–52. **[** P. K. Lopatin, “An algorithm for investigating of an object reachability by a manipulator in an unknown environment,” (in Russian), *Mechatronics, Automation, Control*, no. 9, pp. 49-52, 2012. **]**
5. **Нильсон Н.** Искусственный интеллект. М.: Мир, 1973. **[** N. Nilsson, *Artificial intelligence,* (in Russian). Moscow: Mir, 1973. **]**

ОБ АВТОРЕ

ЛОПАТИН Павел Константинович, доц. каф. информатики и выч. техники. Дипл. инж.-электромех. (Будапештск. техн. ун-т, 1991). Канд. техн. наук (СибГАУ, 1998). Иссл. в обл. интеллектуальной робототехники.

METADATA

Title: A modified forward search algorithm in a problem of a manipulator control in an unknown environment.

Authors: P. K. Lopatin[1]

Affiliation: Siberian state aerospace university named after academician M. F. Reshetnev (SibSAU), Krasnoyarsk, Russia.

Email: efa14@yandex.ru.

Language: Russian.

Source: Vestnik UGATU (scientific journal of Ufa State Aviation Technical University), vol. 18, no. 1 (62), pp. 198-203, 2014. ISSN 2225-2789 (Online), ISSN 1992-6502 (Print).

Abstract: An algorithm for an n-link manipulating robot (MR) control in an environment with unknown static obstacles is considered. The algorithm is reduced to solution of a finite number of problems PI – planning of a path in an environment with discovered and therefore known forbidden states with subsequent execution of the path. Given an algorithm for a solution of the PI problem. A theorem is proved stating that the algorithm will solve the PI problem in a finite number of steps.

Key words: robot; unknown environment; obstacles; reachability.

About author:

LOPATIN, Pavel Konstantinovich, Docent, Dept. of Informatics And Computing Techniques. Dipl. Mech. Engineer (Tech. Univ. of Budapest, 1991). Cand. of Tech. Sci. (SibSAU, 1998).