ROBOT MOTION PLANNING BY APPROXIMATION OF OBSTACLES IN CONFIGURATION SPACE

Martin Ruehl, Hubert Roth

Institute of Automatic Control Engineering
University of Siegen, Hoelderlinstr. 3, 57068 Siegen, Germany
martin.ruehl@uni-siegen.de

Abstract: This paper describes an approach for automatic robot motion planning. A starting path consisting of the direct line connection from start to goal in the configuration space is generated first. At the most colliding configuration a new point is added and moved out of the collision. For this the configuration space is approximated using a corresponding skeleton robot and a point-like obstacle. A collision free path for this approximation is generated by using the inverse kinematics of the skeleton robot. Then the algorithm is started again. By this the whole path is moved stepwise out of the collision.

Keywords: robotics, robot programming, robotic manipulators, industry automation, obstacles, obstacle avoidance, path planning, configuration space

1. INTRODUCTION

Before a robot manufacturing cell is built up in a real production hall it is simulated first within a robot simulation tool. Here the placement of the robots in the environment can be planned and it can be tested, whether the robots are able to reach all desired points in the cell. These points should of course be reached without any collisions. At present this motion planning process is still done by the users of the simulation system. They are forced to generate paths and to test them for collisions. The paths must be varied and tested again, until they are collision free. This procedure is very time consuming and expensive. Existing path planning algorithms are often not used, because a lot of parameters for these algorithms have to be set right depending on the actual planning problem to get the algorithms working well.

Nevertheless there are lots of algorithms available for automatic robot motion planning in literature (see e.g. in Gupta, Kamal K. (1998); Hwang, Y. K., Ahuja, N. (1992); Latombe, Jean-Claude (1991); Lozano-Peres, T. (1986)). Almost all use the so called configuration space. Most of these algorithms need to determine the colliding areas within the configuration space in a kind of precalculation before starting the planning phase. After this time consuming precalculation the information is used for a fast path planning. But unfortunately all generated information is worthless, after something has changed in the robot cell, e.g. an obstacle has moved to another position. In this case the precalculation has to be done again. Another problem of many algorithms in the literature is the use of randomized search. By this it is possible that the user will get two completely different paths, if the algorithm is executed twice, without changing anything.

The algorithm presented here does not need any kind of precalculation before the planning phase. The user will always get the same solution, if the system is used more often with the same conditions. The algorithm determines the most colliding configuration at the direct line connection between the start and the goal configuration. Then an approximation of the obstacle within the configuration space is developed by the use of a corresponding skeleton robot and a point-like obstacle. A collision free path for this approximation is generated by using the inverse kinematics of the skeleton robot. Tangents at the approximated obstacle are determined and used for a collision free path around this obstacle. Hereafter the algorithm checks

the generated path for further collisions. By this the whole path is moved stepwise out of the collision. Further tests show that this algorithm is able to generate collision free paths for robots with many degrees of freedom in a very short time period.

2. THE CONFIGURATION SPACE

The main idea of the configuration space is to represent the robot as a single point in an appropriate space and to map the obstacles in this space. By this the problem of motion planning for a dimensioned body is transformed into the problem of planning the motion of a single point.

A position of a manipulator robot can be completely described using the values of all joints of the robot at a particular robot position. This list of joint values is called a configuration. All possible configurations build the configuration space. The number of joints of the robot is equal to the dimension of the configuration space. Figure 1 demonstrates the correlation between a robot's position and the corresponding point in the configuration space for a planar manipulator robot with two degrees of freedom.

In the configuration space all configurations which lead to a collision between the robot and at least one obstacle in its environment can be marked. This can be done pointwise by moving the robot in the simulation to every possible configuration and performing a collision check at this position. Thereby the obstacles are mapped from the workspace into the configuration space. Unfortunately there is no direct and easy way to do this mapping. It is very difficult to calculate the shape of the obstacles in the configuration space directly from the position and shape of the obstacles in the workspace for every type of robot, because the kinematic chain and geometry of the robot has to taken into account. In addition, as the dimension of the configuration space grows up exponentially with the number of joints, it is not possible to do a collision check for every configuration for robots with a high number of joints. In Figure 2 the mapping of two obstacles is shown for a robot with only two joints.

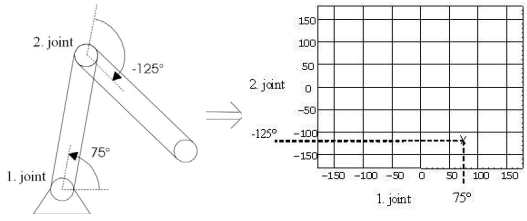

Fig. 1. Correlation between the robot's position and the point in the configuration space

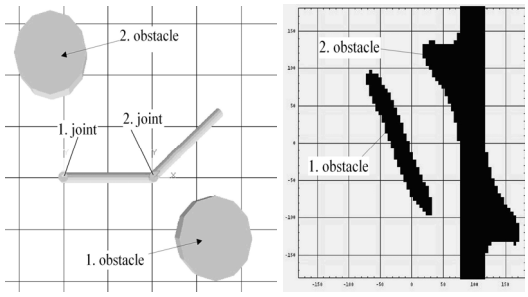

Fig. 2. Workspace and configuration space of a robot with two obstacles and two degrees of freedom

3. STATE OF RESEARCH

There are a lot of algorithms available for robot motion planning in literature. Almost all use the configuration space described above. A good overview of these algorithms can be found in the book of Jean-Claude Latombe (1991). He describes three different types of robot motion planning algorithms.

One type is the so called roadmap method. In this approach path planning consists of capturing the connectivity of collision free areas in the configuration space by developing a network of collision free curves. These curves build a set of standardized paths which is called roadmap. After this roadmap has been constructed path planning is reduced to connecting the initial and the goal configuration to points of this set of paths and to searching within the roadmap for a connection between these points. There are several different methods available for developing the roadmap, e.g. visibility graph, Voronoi diagram or randomized methods. The advantage of this type of path planning algorithm is a fast planning after the roadmap has been developed. The disadvantage is a high computational effort to develop the roadmap. And after changing the position of at least one obstacle, the whole roadmap has to be developed again.

Another type of robot motion planning algorithm is the cell decomposition method. Here the configuration space is decomposed in simple regions, called cells, so that a collision free path between any two configurations within a cell can easily be generated. The adjacency relation between these cells is then searched and stored. The outcome of the search is a sequence of cells from the initial to the goal configuration from which a continuous path can be computed.

The third type of algorithms described in (Latombe, 1991) is the potential field method. An artificial potential produced by the goal configuration and the obstacles is introduced. The goal has an attractive and the obstacles have a repulsive potential. The robot moves in the direction of the gradient of this potential field. The main problem of this method is the existence of local minima in the potential field.

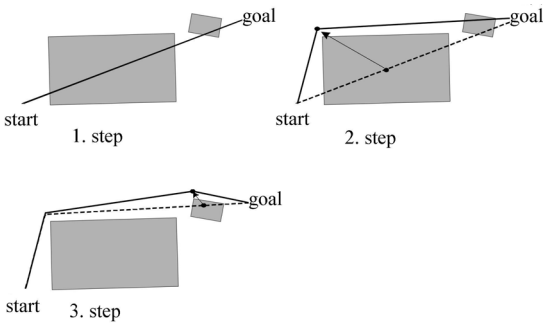

Fig. 3. Example for the basic principle of presented algorithm

4. DEVELOPED ALGORITHM

The algorithm described in this paper does not require any kind of precalculation. Collision checks are only performed at interesting configurations and there is no need to calculate the whole configuration space. Therefore this algorithm can be used for robots with a high number of joints. For one planning phase the robot should be the only moving device. But between two runs of the motion planner the obstacles can be moved around in the workspace and the planning can be started again without additional calculations.

4.1 Basic Principle

The basic idea of the presented robot motion planning algorithm is to generate an initial path between the start and the goal configuration and to vary this path until it is collision free. The initial path consists of the direct line connection from start to goal in the configuration space. If this path is collision free, the algorithm has already found a solution and can stop planning. In general the initial path is of course not collision free. In this case the algorithm searches the "most colliding configuration". A description of this operation can be found in the next chapter.

This configuration is moved out of the collision and added to the path. The resulting path consists of the start configuration, the added and moved point and the goal configuration. In the next step the algorithm checks the complete path for possible collisions again. In case of further collisions the loop will be started again. By this the algorithm moves the path step by step out of the collision. In Figure 3 a simple example for the basic principle of the presented algorithm can be seen.

4.2 Determination of most colliding configuration

As described before the presented algorithm has to determine the most colliding configuration. This is done by using a rating function for the "badness" of a collision. In reality all colliding configurations are just colliding and cannot be rated. But by using the geometry models within a simulation tool it is possible to calculate virtual properties of colliding configurations and compare them. The used rating function was introduced by Boris Baginski. A more detailed description can be found in (Baginski, 1998).

The rating function does not measure the intersecting of the robot and the obstacle but the size reduction required for the robot's geometry model to avoid the collision. In the first step the first colliding link – coming from the base of the robot – is detected. For manipulator robots it is obvious that a collision gets worse if the collision occurs at a link that is nearer to the base of the robot.

In the second step all links that are behind the first colliding link are neglected. The first colliding link is then reduced or scaled down to its joint, until the rest of the robot is no more colliding. Figure 4 gives an example. The relative size reduction of the robot can be used as a rating value for the corresponding configuration. This value can vary between "zero" for a completely downscaled link to "one" for its normal size.

In this way all configurations of a path segment have to be rated. This can be done either by using a discretization of the path segment, or by using the swept volume of the robot. A detailed description of this path segment rating can be found in (Baginski, 1998). After rating all configurations, the most colliding configuration of a path segment can be found. It is the configuration at which the robot has to be scaled down mostly. Here the scaling value is the smallest within the path segment. This configuration has to be moved out of the collision. For this movement it would be very useful to know the shape of the obstacle in the configuration space. But as described before it is very difficult to calculate this shape directly from the geometry of the obstacle in the workspace. One alternative is to use not the exact shape but an approximation and to find a collision free path around this approximated obstacle within the configuration space. The construction of this approximation is described in the next chapter.

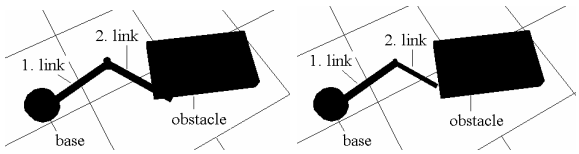

Fig. 4. Example for scaling down the robot size to get the robot collision free

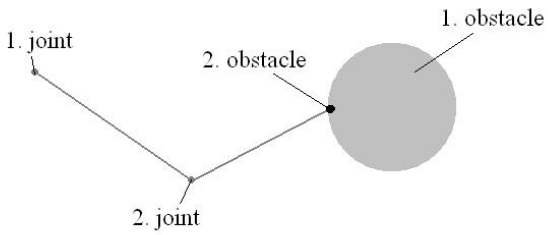

Fig. 5. Approximation of 1. obstacle by the point-like 2. obstacle within the workspace

4.3 Approximation of the obstacle in configuration space

For an approximation of the shape of an obstacle within the configuration space the robot itself is approximated first. The geometry of the robot is reduced to a skeleton geometry. This means that the reduced robot consists only of the base points of each link and the connection line between these points. The links behind the first colliding link are neglected. An additional joint is added to the robot's kinematics which describes the scaling value. In Figure 5 the reduced robot for the two-dimensional example can be seen.

In the next step the reduced robot is set to the most colliding configuration which was determined before. The additional joint is set to the scaling value that corresponds to this configuration. At the end of the kinematic chain of the skeleton robot a point-like obstacle is then created. This obstacle is used as an approximation for the original one. Figure 5 illustrates this in the robot's workspace. The added point-like obstacle is labelled with "2. obstacle" in this picture.

By this construction the added obstacle is placed in a way that the collision with the original obstacle is almost completely avoided, if the robot is able to circumvent this point-like obstacle. At least the scaling value for the examined path segment and the original obstacle's and robot's geometry will be improved.

In Figure 6 the shape of the original obstacle in the configuration space for the example of Figure 5 can be seen. In addition the shape of the added point-like obstacle is illustrated in this picture. This shape is a nonlinear curve in the configuration space which has an inflection point at the centre of the original obstacle's shape and is clinging to the boundary of this shape at its endings. As one can see from the picture, if it is possible to find a path from the given start to the goal configuration which does not intersect with this curve, this path would be almost collision free, i.e. it does not intersect with the shape of the original obstacle. In the next chapter the construction of this path is described.

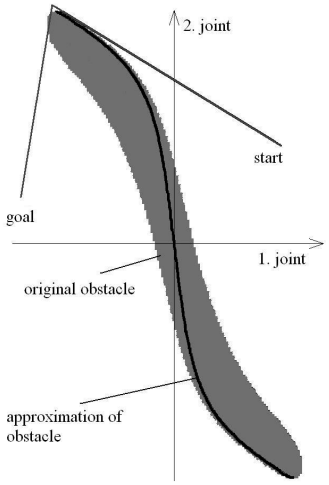

Fig. 6. Approximation of the obstacle within the configuration space by the black line

4.4 Determination of a collision free path around the approximated obstacle

For a given robot the forward kinematics can be calculated by

$$x = f(q) \qquad (1)$$

In this equation the vector x describes the position and orientation of the end of the kinematic chain, the so called tool centre point (TCP). The vector q is a list of all joint values of the robot and $f(q)$ is a nonlinear function of q. This function can be determined for every robot with an open chain kinematics by using the D-H-parameter (Denavit and Hartenberg, 1955).

The equation (1) can also be used to determine the position of the point-like obstacle described above. In this case x contains only a position-vector, as a single point has no orientation. The vector q consists of all joints of the skeleton robot including the added joint for the scaling value (see chapter 4.3). Thus, the vector q has always a higher dimension than the position vector x. This means that the skeleton robot used here is always redundant, i.e. the robot has an infinite number of configurations q which yield to the same position vector x.

To determine a collision free path around the point-like obstacle we need the inverse function of equation (1) to be able to calculate different joint values of the robot from the position of the obstacle. This equation is given by

$$q = f_{inv}(x) \qquad (2)$$

The determination of this equation is a well researched problem in the field of robotics and called the inverse kinematics. Especially for redundant robots this is a nontrivial problem as the inverse kinematics has no single solution but a solution subspace within the configuration space. Several methods have been developed to solve this quite complex problem. An overview can be found in (Nakamura, 1991).

One very often used method to solve the inverse kinematics problem is to use not equation (1) directly but their derivative given by

$$\dot{x} = J(q)\dot{q} \tag{3}$$

Here *J(q)* is the so called Jacobian matrix which describes the movement of the vector *x* for small changes of the vector *q*. Equation (3) is therefore a linearization of equation (1) at a given configuration. For non redundant robots (3) can be rewritten to

$$\dot{q} = J^{-1}(q)\dot{x} \tag{4}$$

With a numerical integration (4) yields to a solution for equation (2). Unfortunately the inverse matrix of *J(q)* does not exist for redundant robots as in this case the matrix is not of quadratic form. The general form of (4) which is also valid for non quadratic Jacobian matrices is

$$\dot{q} = J^{+}\dot{x} + (I - J^{+}J)v \tag{5}$$

In this equation J^{+} is the so called Moore-Penrose Pseudoinverse of the Jacobian matrix. This is the generalized matrix inverse for a non quadratic matrix. The matrix *I* in (5) is the identity matrix and *v* is a free eligible vector. Equation (5) consists of two parts. The first addend gives only one solution for the inverse kinematics problem. As the pseudoinverse is used this solution is the minimum norm solution. The second part describes the so called self moving manifold of the configuration space. If the joint values of a redundant robot are moved in a direction from this subspace the TCP of the robot will not move at all.

The vector *v* in equation (5) can be chosen freely. In (Lièggeois, 1977) this vector is used to optimize a scalar quality function *h(q)*. The gradient of this function replaces the vector *v* in order to maximize the value of *h(q)*. This yields to

$$\dot{q} = J^{+}\dot{x} + k(I - J^{+}J)\frac{\partial h(q)}{\partial q} \tag{6}$$

which can be used to determine a collision free path for the approximated obstacle. If for the function *h(q)* the scaling value for the most colliding configuration (see chapter 4.2) is used, (6) will maximize this scaling value. By this a configuration can be determined by which the robot will just touch but not collide with the obstacle. As the scaling value was added as the last value of *q*, the function *h(q)* and the gradient can be written as

$$h(q) = q_n \qquad \frac{\partial h(q)}{\partial q} = \begin{pmatrix} 0 \\ \vdots \\ 1 \end{pmatrix} = e_{scale} \tag{7}$$

In (6) $\dot{x}$ describes the movement of the point-like approximation of the obstacle. As the obstacle should not be moved within the motion planning this value can be set to zero. So we come up with an equation that describe the direction in which the robot should move to escape from the collision

$$\dot{q} = k(I - J^{+}J)e_{scale} \tag{8}$$

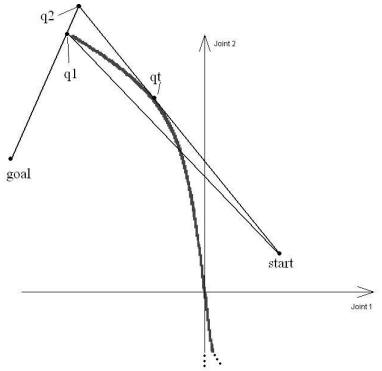

Fig. 7. Collision free path construction using the tangent at the approximated obstacle

Starting from the most colliding configuration, with (8) new configurations can be calculated that improve the scaling value. By this the robot can be moved out of the collision step by step. When the last value of *q* which is equal to the scaling reaches the value "one", the robot is no more colliding with the point-like obstacle. This new configuration can be added to the original path segment in order to create a collision free path around the point-like obstacle.

In many cases the path constructed in a way described above is already collision free corresponding to the approximation of the original obstacle. But in other cases this path is still colliding with the point-like obstacle. In these cases in addition to the collision free configuration with the scaling value of "one" (see above) a tangent at the curve of the approximated obstacle in the configuration space should be taken into account. Figure 7 gives an example. Here the path consisting of the configurations *start*, *q1* and *goal* is still colliding with the obstacle. Therefore a tangent at the obstacle's curve in the configuration *qt* was determined. The configuration *q1* was replaced by *q2* which is the point of intersection between the two lines *start-qt* and *goal-q1*. The resulting path is no more colliding with the point-like obstacle.

To determine the configuration *qt* which is needed for the tangent described above the second part of equation (5) can be used. The matrix *M* given by

$$M(q) = (I - J^{+}(q)J(q)) \tag{9}$$

consists of vectors, which are the partial derivations of the self moving manifold at the configuration *q*. The rank of this matrix

$$r = rank(M) \tag{10}$$

is equal to the degree of redundancy of the robot at this configuration. The direction of the tangent in the configuration *qt* must be part of this self moving manifold. Therefore the difference vector

$$d = start - qt \tag{11}$$

must be linear dependant of the vectors in *M*. Thus, if the matrix *M* is extended by the vector *d*, i.e.

$$M_2 = (M\ d) \tag{12}$$

the rank of this extended matrix M_2 should not increase. That means that both the rank of *M* and the rank of M_2 should be equal

$$r = rank(M) = rank(M_2) \tag{13}$$

Thus, equation (13) can be used to check, if a linear line connection from *start* to a configuration *qt* is a tangent at the curve of the approximated obstacle.

5. EXPERMENTAL RESULTS

Several experiments were made to test the abilities of the described algorithm. The robots were simulated by the robot simulation tool IGRIP on a PC with a Pentium IV Processor with 3 GHz. The algorithm was able to find a collision free path in almost all cases. Two examples are shown below. In Figure 8 a collision free path for a robot with three joints can be seen. There are four obstacles in the workspace of the robot that are partly very close to the base of the robot. The algorithm generated the shown collision free path for this example in about seven seconds.

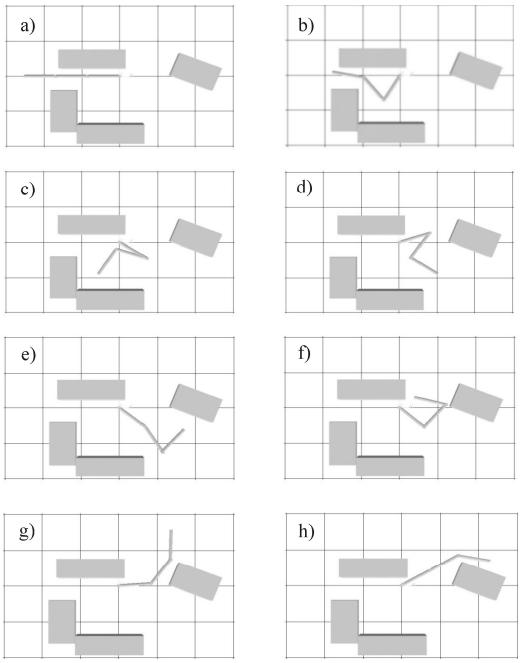

Fig. 8. Example for a collision free path for a robot with three joints

The second example is quite similar to the first one. But here we used a robot with five degrees of freedom. Again the algorithm was able to find a collision free path. For this task the path was generated in about 20 seconds.

CONCLUSIONS

In this paper a new approach for automatic robot motion planning has been presented. It has been shown that the geometry of an obstacle in the robot's workspace can be approximated by a well placed point-like obstacle. After the geometry of the robot has been approximated by a skeleton robot, the shape of this obstacle is computable by using the inverse kinematics of this robot. To determine a collision free path around the approximated obstacle

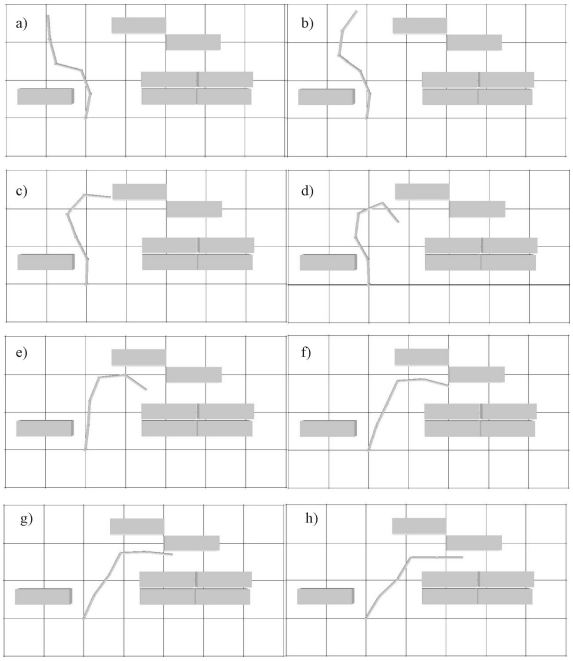

Fig. 9. Example for a collision free path for a robot with five joints

the tangents at the obstacle's shape within the configuration space have been used. This path is mostly collision free for the original obstacle, too. At least the scaling value for the determined path segment has been improved. This algorithm has been successfully tested for different robot motion planning problems.

REFERENCES

Baginski, Boris (1998), *Motion Planning for Manipulators with Many Degrees of Freedom – The BB-Method*, PhD Theses, University of Munich

Denavit J. and Hartenberg R. S. (1955), *A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices*, ASME Journal of Applied Mechanics, pages 215-221

Gupta, Kamal K. (1998), *Overview and state of the art,* Practical Motion Planning in Robotics: Current Approaches and Future Directions, pages 3-8, John Wily and Sons

Hwang, Y. K., Ahuja, N. (1992), *Gross motion planning – a survey.* ACM Computing Surveys, 24(3), pages 219-291

Latombe, Jean-Claude (1991). *Robot motion planning*. Kluwer Academic Publisher, Massachusetts

Liègeois (1977), *Automatic supervisory control of the configuration and behaviour of multibody mechanisms*, IEEE Trans. Systems, Man and Cybernetics, Band SMC-7, S. 868-871

Lozano-Peres, T. (1986), *A simple motion planning algorithm for general manipulators,* Procceedings of national confernence on artifical intelligence AAAI'86, Philadelphia, Pensylvania

Nakamura, Y. (1991), Advanced Robotics, Redundancy and Optimization, Addision-Wesly