May 1992

Fast Motion Planning for Anthropometric Figures with Many Degrees of Freedom

Wallace S. Ching
University of Pennsylvania

Norman I. Badler
University of Pennsylvania, badler@seas.upenn.edu

Fast Motion Planning for Anthropometric Figures with Many Degrees of Freedom

Abstract
Human-like robots are useful in many areas such as deep sea mining and space applications. An efficient motion planning algorithm for these type of robots will be helpful in achieving task level programming. In this paper we present a new efficient algorithm that has successfully computer collision free motions for anthropometric figures with many degrees of freedom within a clustered environment.

Proceedings of the 1992 IEEE
International Conference on Robotics and Automation
Nice, France - May 1992

Fast Motion Planning for Anthropometric Figures with Many Degrees of Freedom

Wallace Ching
Norman Badler
Computer Graphics Research Laboratory
Department of Computer and Information Science
University of Pennsylvania, Philadelphia, PA 19104-6389

Abstract

Human-like robots are useful in many areas such as deep sea mining and space applications. An efficient motion planning algorithm for these type of robots will be helpful in achieving task level programming. In this paper we present a new efficient algorithm that has successfully computed collision free motions for anthropometric figures with many degrees of freedom within a cluttered environment.

1 Introduction

The problem of motion planning for manipulator arms has been a much researched topic. The problem is usually defined as finding a path for the robot arm from a given initial configuration to a final configuration which avoids collisions with the obstacles in the workspace. Many of these algorithms are based on the use of *configuration space*[8] (Cspace) which is the space of the degrees of freedom (DOFs) of the robot.

However, it is well known that the worst case time bound for an exact motion planning algorithm is exponential in the dimensionality of its configuration space. [1] [2]. Direct application of the algorithms will be impractical in cases where the number of DOFs exceeds three. Practical algorithms that make use of heuristics and special search strategies have been devised for robots with relatively few degree of freedoms. [3]

In this paper, we are particularly interested in a class of robots which resemble the human structure. They are characterized by having a torso, two manipulator arms, a head mounted with vision system and also a lower body which is either a transport unit or legs. The total number of DOFs involved are usually fairly large for these robots. This type of robots are usually designed for tasks such as deep sea mining, nuclear plant clean-up, space shuttles operation and bomb handling. Programming at the joint level for these robots will be extremely tedious. We feel that having an algorithm that can plan the motion for this type of robots will be extremely useful in a lot of applications. In this paper we are going to present a novel algorithm that handles this task. Besides, our algorithm can be applied in motion planning for human figure models. This class of models are useful in areas like human factor engineering and manufacturing design. Robots that are hightly human-like are also under development in countries such as Japan. Our algorithm will again be useful in planning motions for these robots.

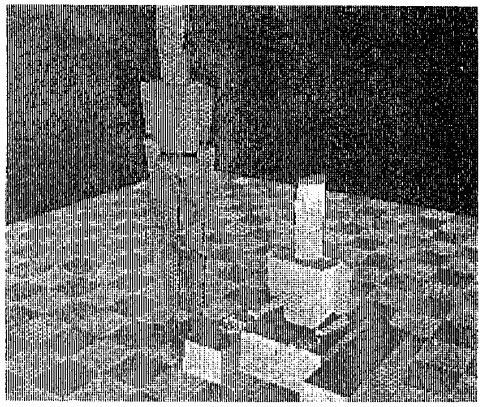

Figure 1: The anthropometric figures we used in this paper

Recently, work has been reported on algorithms that can handle robots with many degrees of freedom. [4][5] [6]. Latombe's approach [4] uses an artificial potential field and an optimization procedure to search for the solution. Gupta's method [5] is a sequential search method.

Our method is closer to Gupta's. However, the fact that anthropometric figures are a tree structure with branching rather than a sequential linkage poses unique challenges that have not been addressed by any existing algorithms.

2 Hierarchical structure of the anthropometric figures

In this paper, we consider two types of anthropometric figures (Figure 1). The first type is modelled after a human being. The second type resembles a bomb discharge robot.

The basic structure of the figures consists of a *torso*, a *head* and two *arms*. In addition, Figure 1 (a) has a pair of legs as the lower body whereas Figure 1 (b) has a *transport unit*.

We will use mainly Figure 1 (b) to explain our algo-

0-8186-2720-4/92 $3.00 ©1992 IEEE

rithm. Extension of the algorithm to Figure 1 (a) will be explained in a later section.

We will present our approach in details in the next few sections.

3 Our Approach

3.1 Overview

The basic idea of the whole algorithm is to first divide up the DOFs of the figure into a number of *Cspace groups* (C groups). Each C group deals with a n-dimensional *configuration space* with the variables being a parameter characterizing the motion of the preceding group and $n-1$ DOFs associated with some linkages.

The first stage of our approach is similar to the sequential search strategy presented by Gupta [5]. However, there are significant differences between ours and his approach. For example, instead of considering one degree of freedom (DOF) at a time, we extend the notion further and consider a *Cspace group* which can involve a variable number of DOFs. In our implementation, we consider two in most cases. In this case, we first plan the path for the first 3 DOFs (q_1, q_2, q_3), parameterizes the path (parameter p), and then plan the path for the next two DOFs in (p, q_4, q_5) and so on. We find that working in a 3-dimensional space is still practical while it also has has the advantage of providing more flexibility and backtracking capability during the planning process and hence a greater chance of finding a collision free path successfully.

We also introduce a *base point* into the figure which has two translational DOFs and one rotational DOF. This will situate at the waist joint and make the lower body of the robot much like a mobile robot with translational capability. As will be seen, these translational DOFs can be handled in a similarly efficient way as the other DOFs in our algorithm.

Moreover, our algorithms are particularly adapted to the hierarchical structure of the anthropometric figure. Many specific problems encountered are addressed by the algorithm and will be explained in later sections. For example, the fact that the anthropometric figure is a tree structure poses unique problems which have to be addressed specificially.

The organization of the planner is as follow: first the *basic algorithm* is then run within each C group to compute a collision free path for the DOFs associated with this group. Next, the *sequential algorithm* is run over the branches and finally a *control algorithm* is run over the whole figure to obtain the final path. After that, a playback routine will be called upon to play back the overall collision free path for the whole figure based on the computed information stored within each of these *cspace groups*. We will cover the detail of these algorithms in the next few sections.

3.2 The grouping scheme

Let us now look at the grouping scheme of the cspace groups before we go into the details of the algorithm. The 1st *cspace group* deals with the three DOFs of the base point. This is the only group that does not contain a parameter. Each succeeding group consists of one parameter which characterizes the motion of the preceding group and a variable number of DOFs (1 or 2 in our implementation). For example, the 2nd group deals with the parameter from the 1st group and two *rotational* DOFs of the waist. The 3rd group handles one parameter and the third rotational DOF of the waist.

Next we come to consider the torso. The fact that the anthropometric figure is not a sequential linkage but rather a tree structure complicates the problem. As can be seen, the torso branches upward into three separate links, namely the head and the two arms. The collision free motion computed for the torso will be considered when planning the motions of all these three links. However, there is a potential conflict here as the three links may come up with a different final motion for the torso. The problem is further complicated by the nontemporal interpretation of the parameter. All these will be addressed in more details in a later section.

The ordering we adopt is to compute the right arm first, then the left arm and finally the head. This is in effect a depth first traversal of the tree structure. Hence, the 4th group handles the torso parameter and two *rotational* DOFs of the right shoulder. The 5th group takes care of the remaining DOF of the shoulder and the right elbow. The 6th and 7th group handles the 3 DOFs of the right wrist. Groups 8, 9, 10, 11 accounts for the left arm and groups 12, 13 for the head.

4 Basic algorithm - computing a collision free path within a cspace group

To compute the collision free path for the DOFs involved in a cspace group, we adopt the approximate algorithm by Lozano-Perez based on a configuration space approach [8] since the resulting algorithm is ready for machine implementation [7]. We will briefly describe the algorithm below with a 3-dimensional C group and highlight some of the design choices that we have made. Interested reader should refer to the details in [7].

First we compute the forbidden regions within this 3-dimensional $t \times \theta^2$ configuration space. The forbidden regions are obtained by first discretizing the legal range of the joint angle up to a predefined resolution, then at each increment, checking for collision with the obstacles in the environment. Areas where no collision can occur are called *free space*.

Next adjacent slices are grouped into *regions* so as to make use of the connectivity relationship and cut down the size of the data. We group the slices in the free space into *regions* as in [7] rather than using *visibility graph* as in [5].

The reason is that path computed from the visibility graphs are inevitably close to the obstacles at certain points. In fact, certain points on the computed path will barely miss the vertices of some obstacles. The fact that the computed path of one group being so close to the obstacles will leave little leeway for the links involved in the succeeding group to maneuver. This will greatly reduce the chance of successfully finding a collision free path for these groups. Therefore, we choose to use paths that are close to the center of the free regions. This will give more space for the links in the succeeding groups to maneuver.

The next step is to build a graph to represent the con-

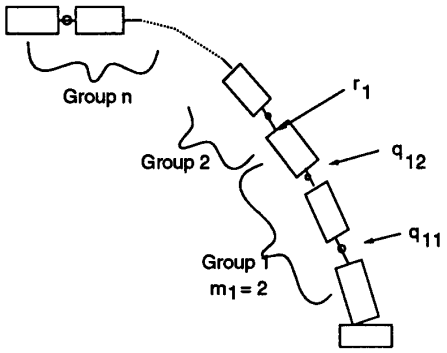

Figure 2: Example of a sequential linkage

nectivity relationship between these regions. Finally, a collision free path is obtained by searching this connectivity graph starting from the node (region) containing the initial configuration to the node containing the goal configuration using an A* algorithm.

5 Sequential algorithm - free path for a sequential linkage

Before we look at the whole tree structure of the anthropometric figures, we first outline how the basic algorithm is applied by the *sequential algorithm* to compute a free path for a sequential linkage. This part is similar to the sequential strategy presented in [5].

Referring to Fig 2, let n be the total number of *cspace groups* on this branch. Let the joint DOFs within the groups be represented by q_{ij} where i is the group number and j is either 1 or 2 since there are either 1 or 2 DOFs associated with each group. Let r_i be the reference vertex for group i. It is basically the far side vertex of the link associated with the DOFs in the group i. Let $r_i(t)$ denote the trajectory of the reference vertex r_i. The initial and goal configurations of the arm are given as q_{ij}^s and q_{ij}^g, i=1,n; j = 1,2.

The algorithm is as follows:

1. Compute a collision free trajectory for the links associated with group 1. The trajectory of the reference vertex on this link will be $r_1(t)$.
2. $i = 2$.
3. While ($i < n$)
   (a) along $r_{i-1}(t)$, compute a collision-free trajectory for the DOFs in the ith group from q_{ij}^s to q_{ij}^g for j = 1,2, using the *basic algorithm*.
   (b) given $q_{1j}(t), q_{2j}(t), ..., q_{ij}(t)$, compute $r_i(t)$ using forward kinematics.
   (c) Increment i.

5.1 Interpretation of the parameter t

The parameter t can either has a temporal [10] or non-temporal interpretation [5]. If the parameter is interpreted as time, it means that the complete motion along the path is determined and that it cannot be changed while planning the motion of subsequent links in subsequent groups. On the other hand, if the parameterization of the path is non-temporal, then only the path is determined. How point r_i moves along this path is determined while planning the motion of subsequent links. This means that backup movement is allowed under this interpretation.

We have chosen to use the nontemporal interpretation of the parameter in most cases since we found that the backup movement is crucial in finding a collision free path in most tested environment. The fact that the collision free path computed for a particular group depends on the path from the preceding groups has already constrained its chances of finding a path. Adding the extra constraints that the parameter t being temporal will leave very little room for the algorithm.

6 Control algorithm - computing a collision free path for the whole figure

Now that we have considered computing a collision free path for a sequential branch, we can go further and consider the whole tree structure. The basic idea of the algorithm is to start from the very first group, compute its free path, parameterize the resulting motion, then repeat the procedure for the rest of the body following a particular traversal order.

Since there are three branches coming out from the torso, we have to impose an ordering in the groups when computing the free path. We choose to compute the free path for the right arm first, then the left arm and finally the head branch. This is in effect executing a depth first traversal of the tree structure.

Finally a special playback routine is needed to traverse the tree, collect the path information computed and playback the final coordinated collision free motion.

Note that after we find the free path for the whole right arm, we have to record all the joint angles along this path. The information is important in the final stage of playing back the free path.

There are some specific issues needed to be addressed in each of these stages. These are explained in details in the following sections.

6.1 Computing a collision-free path for the very first group

The first C group differs from all the other groups in that it is the very first group and therefore deals with DOFs only. The two translational DOFs and one rotational DOF make the linkage of the this group behave just like a mobile robot. However, the linkage associated with this group is not well defined. It can be either Figure 3 (b),(c) or (d). The difference lies in what kind of end result motion you would like to have. Figure 3 (b) has the torso size equal to the real torso. The path computed for this *linkage* will be collision-free for the whole torso in this orientation. Hence, no torso movement will be needed and only arm movements may be necessary along this path. Figure 3 (c) has the torso size equal to the thickness of the real torso in its upright position. The path computed

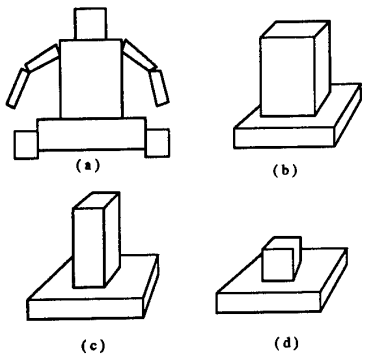

Figure 3: Possible linkages to be associated with the first cspace group

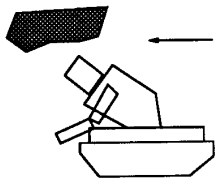

Figure 4: The figure has to bend forward to avoid obstacles from above.

for this linkage may require some rotation of the torso in tight situations. Figure 3 (d) has the torso size to be equal to the minimum dimension of the real torso. The path computed may require lots of torso movement to fully guarantee collision free motion at some tight places. For example, at a certain part of the path the torso may actually have to bend forward to fully avoid the obstacles. (Figure 4)

6.2 Resolving the conflicts between different branches

Although the three branches are attached to the same rear part of the torso, we do not use the same parameter t that parameterize the motion of the torso in computing all these branches.

This is due to the fact that we have allowed the parameter t to be interpreted as a nontemporal parameter which again implies that backtracking is allowed and the values of t along the computed path may be nonmonotonic. If we use the same parameter of t for all the first groups in these branches, the corresponding joint angle values cannot be computed uniquely during the final playback phase.

Our solution to this problem is to further *parameterize* the already *parameterized path* of the torso. Let us look at Figure 5 (a). After we have computed motion for the torso, we parameterize the resulting path with the parameter, say t_1. This is the track shown attached to the torso. Next, we compute the path for the first group of the right arm based on this parameterized path. For illustrative purpose, we assume the right arm has only one C group. The resulting configuration space for this C group

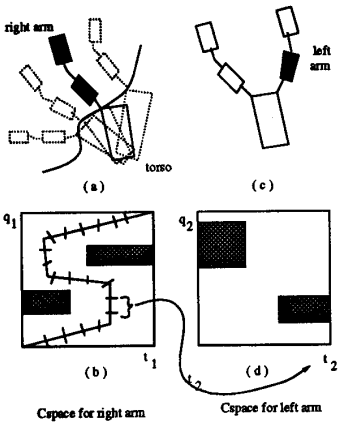

Figure 5: (a) the parameterized path of the torso. (b) the Cspace and computed path for the C group of the right arm. t_2 is the parameter along this path. (c) the left arm is highlighted. (d) the Cspace and computed path for the left arm

and the free path may look like the one in Figure 5 (b).

As we follow the computed path, it can be seen that the values taken by the parameter t_1 is not monotonic. We choose to parameterize this path again before going on to compute the path for the left arm. We will call this parameter along the path t_2 and it is this parameter that we are using when dealing with the first cspace group of the left arm. This is shown in Figure 5 (d).

The reason for this extra step will become clear after we have shown how the overall motion is played back.

6.3 Playing back the free path

After computing the free path for all the linkages, we need a special routine to coordinate all the computed information and play back the overall collision free path.

For example, let Figure 6 represent the configuration space for the last group of the *left* arm (2-dimensional space for illustrative purpose). We then discretize the free path according to a particular resolution. At every discretized point, say A, there is a corresponding (θ, t) pair standing for the actual angle we should set the corresponding joint DOF to, and also a *parameter* t referring to the motion of the *preceding* group. We first set the DOF to the θ value. Then we use the parameter to trace back to the *preceding* group. Note that within this preceding group, the parameter t is *monotonic*. Hence we can uniquely determine the corresponding (θ, t) pair within this group. With the same token, we can trace back to the group further preceeding this one. We carry on in this fashion recursively until we encounter the parameter that accounts for the motion of the *first* group of the *right* arm. Note that at this point, all joint DOFs of the left arm have already been set to their correct value.

The whole sequence of the right arm movement should have been recorded at this point. The parameter we are now considering is the same as t_2 described in the last section. As discussed above, it is monotonic in values. Hence, we can uniquely determine the set of angles cor-

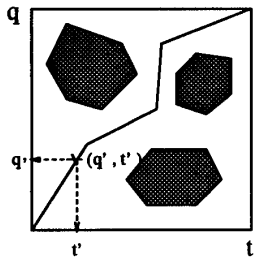

Figure 6: A sample C space for the last group in the left arm illustrating the playback mechanism

responding to the *right* arm DOFs. This is exactly the reason why we have to parameterize the path of the torso again as we need to have a variable which is monotonic and so can uniquely determine the angles it is associated with.

After setting the angles in the right arm, the torso will be set in a similar fashion.

7 Extension to the Human Model

The above algorithm can be easily extended to run on the human figure as in Figure 1(a). For a human figure, the movement of the base point is no longer restricted to move on a planar surface. Hence, we need to introduce another translational DOF to the *base point*. We may need one more rotational DOF too depending on what kind of motion we want the figure to have.

The two legs of the human figure can be treated as two additional links. Since they both connect directly to the lower torso (a linkage associated with the base point), they can be applied with the sequential algorithm separately. There is no complication involved here.

However, only leg movement within a stepping cycle can be simulated. Leg movement that require preriodic locomotion cannot be generated by this algorithm.

8 Results and discussion

We have tested our algorithm on a number of cases to a certain level of success. Figure 7 shows the same figure reaching through two holes with two arms. These examples run within a graphical environment named Jack[9] on a Silicon Graphics Personal Iris workstation. The initial and final configuration of the figures are specified by manually adjusting the figures positions. An interactive technique based on inverse kinematics has greatly simplified the task [11]. It takes about 9 minutes wall clock time to compute this example.

Figure 8 shows a second example in which the bomb handling robot navigates an obstacle cluttered environment. This is a more difficult test case. We have to finely adjust the parameters like distance of free path from obstacles and the resolution in order to obtain a reasonable solution. However, the resulting path are still somewhat awkward and unnatural. We are currently working on postprocessing technique to smooth out the resulting motion.

The fact that we have chosen the solution to be a path that is near the center of the free space often creates the visual effect that the figure is moving in a path which is farther from the obstacles than necessary. The distance of the free path from the forbidden region in the configuration space can be set as a variable. The user can first run the system in default to get a solution and then decrease this variable to get a more optimal path.

As for the complexity of the algorithm, since the number of DOFs in a group is upper bounded, the run time for the *basic algorithm* can be treated as a constant. Hence the run time of the whole algorithm without backtracking is $O(p)$ where p is the total number of DOFs. With backtracking, the worst case is that we have to search through the whole tree and the run time become exponential.

When dealing with a human figure, the notion of *naturalness* will come into consideration. It is still not known how to quantify natural behavior. Besides, simulating the walking mechanism by itself is a topic requiring more investigation and is not readily available for our system at this point.

Acknowledgements

This research is partially supported by Lockheed Engineering and Management Services (NASA Johnson Space Center), MOCO Inc., NSF CISE Grant CDA88-22719, and ARO Grant DAAL03-89-C-0031 including participation by the U.S. Army Human Engineering Laboratory, Natick Laboratory, TACOM, and NASA Ames Research Center.

References

[1] J. T. Schwartz, M. Sharir, "On the Piano Movers problem I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers", Rep. No. 39, Dept. Comp. Sci. Courant Inst. Math. Sci., New York, NY, 1981.

[2] J. T. Schwartz, M. Sharir, "On the Piano Movers problem II. General techniques for computing topological properties of real algebraic manifolds" , Rep. No. 41, Dept. Comp. Sci. Courant Inst. Math. Sci., New York, NY, 1982.

[3] B. Donald, "Motion planning with six degree of freedom", MIT AI Tech. Rep. 791, pp. 504-512, 1984.

[4] J. Barraquand, B. Langlois, J. Latombe, "Robot Motion Planning with Many Degrees of Freedom and Dynamic Constraints", *Fifth International Symposium on Robotics Research, (ISRR) Tokyo, Aug 1989*

[5] K. K. Gupta, "Fast Collision Avoidance for Manipulator Arms: A Sequential Search Strategy", *IEEE Transactions on Robotics and Automation, Vol.6, No. 5, Oct. 1990.*

[6] B. Faverjon, P. Tournassoud, "A Practical Approach to Motion-Planning for Manipulators with Many Degrees of Freedom", International Symposium of Robotics Research, 1990.

[7] T. Lozano-Perez, "A Simple Motion-Planning Algorithm for General Robot Manipulators", *IEEE Journal of Robotics and Automation, Vol RA-3, No.3, June 1987*

[8] T. Lozano-Perez, "Spatial Planning: A Configuration Space Approach", *IEEE Transactions on Computers, Vol. C-32, No. 2, February 1983*

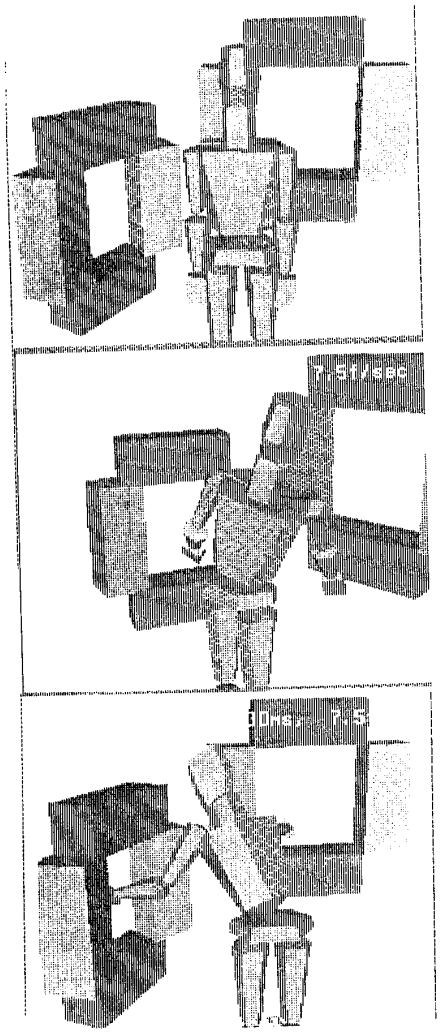

Figure 7: A human figure reaching through apertures using two arms

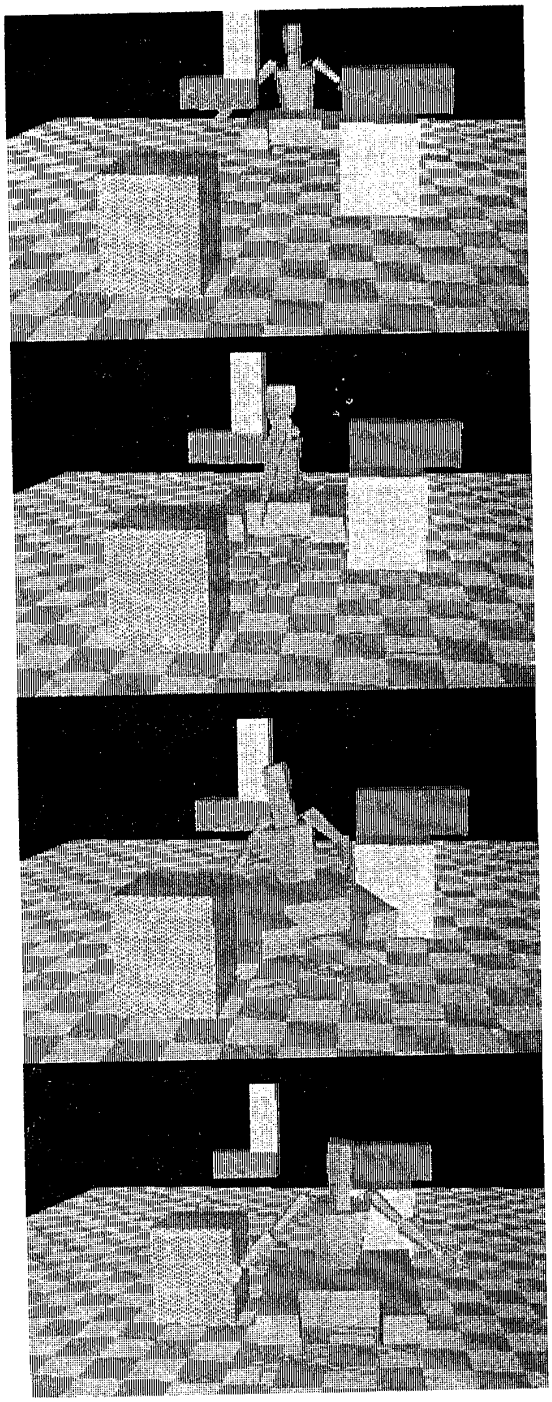

Figure 8: A bomb discharge robot navigating an obstacle cluttered environment

[9] C. Phillips, N.I.Badler, "Jack: A toolkit for manipulating articulated figures", ACM/SIGGRAPH Symposium on User Interface Software, Banff, Canada, October 1988.

[10] K. Kant, S. Zucker, "Toward Efficient Trajectory Planning: The Path-Velocity Decomposition", *Internal Journal of Robotics Research, 1986 Fall.*

[11] C. Phillips, J. Zhao, N.I. Badler, "Interactive real-time articulated figure manipulation using multiple kinematic constraints", *Computer Graphics* 24(2), pp. 245-250, 1990.