Motion Planning of Multi-Joint Robotic Arm with Topological Dimension Reduction Method

Bo Zhang, Ling Zhangt , Tian Zhang

Dept. of Computer Science
Tsinghua University
Beijing, China

"f Dept. of Mathermatics
Anqing Teachers' College
Anhui, China

ABSTRACT

This paper explores the realization of robotic arm motion planning, especially Findpath Problem, which is a basic motion planning problem that arises in the development of robotics. Findpath means: Given the initial and desired final configurations of a robotic arm in 3-dimensional space, and given descriptions of the obstacles in the space, determine whether there is a continuous collision-free motion of the robotic arm from the one configuration to the other and find such a motion if one exists. There are several branches of approach in motion planning area, but in reality the important things are feasibility, efficiency and accuracy of the method. The topological method has shown great potentiality in practice compared with the others. Here a simulation system is designed to embody the topological Dimension Reduction Method (DRM) [I]-[2] and it is in sight that DRM can be adopted in the first overall planning of real robotic arm system in the near future.

1. INTRODUCTION

So far there are three main branches of approaches to motion planning. They are the geometrical approach, which is represented by the Subdivision Method [6]; the expert-system-based approach, which is very uncertain, or task-dependent; and the topological approach, which is presented in [1],[8]. The same point of them is to map the physical space to the Configuration Space (C-Space) [5], which is actually the state-parameter space of the robotic arm. For example, suppose the robotic arm has n degrees of freedom, then the C-Space of it will be $\Theta_1 \times \Theta_2 \times \cdots \times \Theta_n$. With the mapping, a state of the robotic arm in physical space corresponds to a point in the C-Space; a physical obstacle corresponds to a C-Space-Obstacle, which means the obstacle is expressed with the C-Space coordinates. So the Findpath Problem in physical space is equivalent to the investigation of C-Space. Usually, C-Space is a high-dimensional space. To study the connectivity of such a high-dimensional space, the topological Dimension Reduction Method presented in [1] is superior in many aspects.

2. DIMENSION REDUCTION METHOD

Because the mathematical proofs and deductions about DRM [2] are very time-consuming and abstract, here we only outline the main idea of DRM. What is emphasized in this paper is to demonstrate how a computer realization is derived from the method.

In C-Space, the set made of points corresponding to collision-free states of a moving rigid object A among obstacles $\{B_i\}$ (i=1...M) is called Free-C-Space (FCS), therefore the set of C-Space-Obstacles is C-Space / FCS, where / is the set difference.

Considering FCS as a graph in C-Space, the motion planning of A is equivalent to the connectivity investigation of FCS. Usually, FCS is a high-dimensional graph. For a 3-dimensional moving rigid object A among obstacles, if x represents the coordinates of a reference point P of A and θ represents the three Euler angles indicating the orientations of the reference axis PQ of A, then FCS = G(F) = {(x, F(x)) | x ∈ D(F)}, where F: $X \rightarrow 2^{\Theta}$ (Θ is the range of possible members of the Euler angles.) is called the Rotation Mapping and $D(F) \subseteq X$. G(F) is a graph in C-Space $X \times \Theta$ and is called the Rotation Mapping Graph (RMG). For a moving rigid object A among polyhedron obstacles, the Rotation Mapping F is always a Semi-Continuous multi-valued mapping [2].

Now we introduce DRM in the form of theorems:

Theorem 2.1: Assume C-Space is $X \times Y$, RMG is $\{x, F(x) \mid x \in D(F)\}$, where F: X→Y is a Semi-Continuous mapping and D(F) is the domain of F, if D(F) is connected , F(x) is compact, and $\forall x \in D(F)$, F(x) is a connected set on Y, then $G(F)=\{x, F(x) \mid x \in D(F)\}$ is a connected set on $X \times Y$.

This conclusion leads the connectivity investigation of a high-dimensional space to that of lower-dimensional spaces. With the requirement of Semi-Continuous the DRM can be extended to a general space $X = X_1 \times X_2 \times \cdots \times X_n$, where X_i (i=1...n) is a sub-space of X. That means the connectivity investigation of X can be dissolved into those of X_i (i=1...n).

In order to fulfill the conditions of theorem 2.1 and then apply the principle to the motion planning problem, the domain D(F) has to be divided into sub-domains $\{D_i(F)\}$ (i=1...r), where r is determined by the square of the obstacle number. Of course every $D_i(F)$ must be a connected set, and also for each $D_i(F)$, $\forall x \in D_i(F)$, F(x) has the same combination of connected branches, noted as $F(D_i) = \bigcup_{j=1}^{n_i} F_j(D_i)$, which means for all x in $D_i(F)$, $F_j(D_i)$ is a connected subset of $F(D_i)$. $F(D_i)$ is the set made of all possible orientation angles on D_i. See Fig 2.1, if the moving object A is a R-long rod PQ among obstacles $\{B_i\}$ (i=1...3) and P is selected as the reference point, the possible connected branches of PQ at x are $F_1(x)$ indicated as (B_1, B_2), $F_2(x)$ indicated as (B_2, B_3), $F_3(x)$ indicated as (B_3, B_1) and finally all combine to indicate F(x) (all possible orientation angles at x).

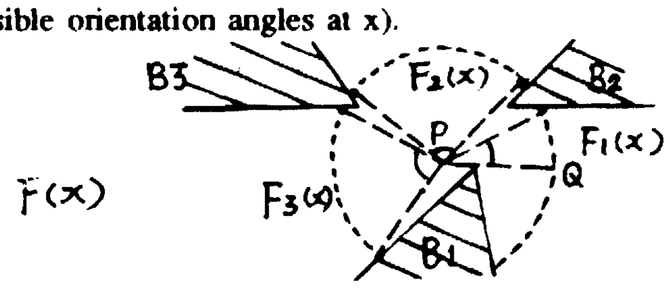


Fig 2.1 F(x) is the combination of three connected branches of x, $F_1(x)$, $F_2(x)$ and $F_3(x)$.

It can be proved that every $(D_i(F), F_j(D_i))$ indicates a connected block on $X \times Y$. This implies that every inner point of $(D_i(F), F_j(D_i))$ is equivalent in the connectivity representations. So we don't bother to deal with the actual value of every point and the whole continuous FCS, but to deal with the common combinatorial representation of every connected block $(D_i(F), F_j(D_i))$ and the

connectivity among these limited number of connected blocks $\{D_i(F), F,(F_j),)\}$. The continuous infinitely-divisible space of real, FCS, is at last partitioned into finite number of equivalent connected blocks. With this kind of abstract evolution, the planning can be realized accurately and efficiently.

There is something to do before we can see its feasibility. First we must answer how to divide D(F) into *{Dj(F)* to adapting to the requirements. By close research on the problem we get to the following partition rule:

Theorem 2.2: D(F) is divided into {D,(F)| with the critical curves which consists of three kinds of curves: R-Growing Boundaries [2], Disappearance Curves (DC) [1],[4] and original obstacle boundaries. The critical curves can be expressed by segmental parametric equations.

See Fig 2.2(a),(b), R-Growing Boundary is produced by expanding the original obstacle boundary by R distance. Disappearance Curves emerge where the convex obstacles are crowded together or some concave obstacles exist. (A concave obstacle can be divided into several convex obstacles.) Disappearance means moving into the DC-region some connected branch will disappear. These are the most interesting curves. In a planar polygon environment, there are only two types of DCs. Assume PQ is a R-long line-segment and P is the reference point, then one type of DCs are generally curves traced by P as PQ rests against one obstacle's comer and another obstacle's edge, which means a DC is generated by a point and a line-segment. See Fig 2.2 (a), within the region closed by the DC, the connected branch $(B_1\ B_2)$ disappears. We can see one type of DC is a conchoid curve or a part loop of a conchoid curve, whose equation is readily calculated. Another type of DC is very simple. See Fig 2.2 (b), it is an arc caused by a concave comer.

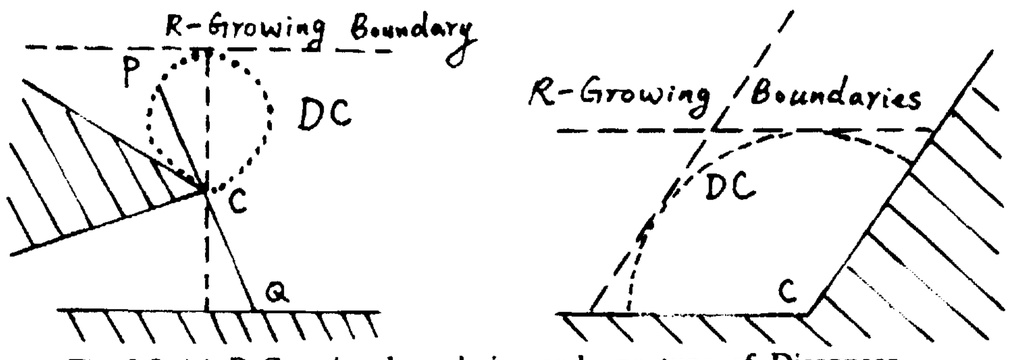


Fig 2.2 (a) R-Growing boundaries and one type of Disappearance Curve

Fig 2.2 (b) R-Growing boundaries and another type of Disappearance Curve

Second how to judge the connectivity of any two connected blocks? Also by applying the principle of DRM, we give a rule:

Theorem 2.3: For any two connected blocks $(D_i(F), F_j(D_i))$ and $(D_k(F), F_l(D_k))$, if the domains $D_i(F)$ and $D_k(F)$ are adjacent and the intersection of the connected branches $F_j(D_i)$ and $F_l(D_k)$ is non-empty, then the blocks $(D_i(F), F_j(D_i))$ and $(D_k(F), F_l(D_k))$ are connected.

Finally simplify each block $(D_i(F), F_j(D_i))$ as a Small-Node, $(D_i(F), F(D_i))$ as a Big-Node, which may contain more than one Small-Nodes for there may be more than one connected branches in $F(D_i)$. If the blocks $(D_i(F), F_j(D_i))$ and $(D_k(F), F_l(D_k))$ are connected then the two corresponding Small-Nodes are linked with an arc. Two Big-Nodes are linked if any of their Small-Nodes are linked. It is obvious that the Small-Nodes belonging to the same Big-Node can't be linked. So the Characteristic Network (CN) [1]-[4] {Nodes,Arcs} has been constructed, which is a 2-step network. The hierarchical constructions again bring about efficiency. Locating the initial and desired final configurations to two Small-Nodes in CN and searching a way between the two Small-Nodes in CN with some graph-searching algorithm of graph theory, we can get a concrete motion planning if and only if there exists a connected way between the two Small-Nodes in CN because CN is homotopically equivalent to RMG.

Here is an outline of the solution of Findpath problem with

DRM:

1. Partition of RMG

D=the domain of the free space on X;

PARTITION D into $\{D_i\}$ (i=1...r) by R-Growing Boundaries, Disappearance Curves and obstacle boundaries;

for all D_i of $\{D_i\}$ do

$F(D_i)=\bigcup_{j=1}^{n_i} F_j(D_i)$= Connected Branches corresponding to D_i;

RMG=$\{D_i, F_j(D_i)\}$ (i=1...r, j=1...n_i);

2. Construction of CN

Big-Node[i]=$(D_i, F(D_i))$;

Small-Node[i,j]=$(D_i, F_j(D_i))$;

for all Small-Node[i,j] and Small-Node[k,l] do

if CONNECTED($(D_i, F_j(D_i)),(D_k, F_l(D_k))$) then SMALL-LINK(Small-Node[i,j],Small-Node[k,l]);

BIG-LINK(Big-Node[i],Big-Node[k]);

3. SEARCH FOR PATH

S=LOCATE(initial state,CN);

G=LOCATE(final state,CN);

Way=SEARCH(CN,S,G);

Path=TRANSFORM(Way);

The main features :

1. It solves the problem in a point of global view, just like man doing.
2. It plans, instead of continuously or numerically, topologically and then combinatorially. As a result, it gains efficiency.
3. It is an accurate, or complete, algorithm theoretically, which meaas it can find the path if and only if one exists.
4. Not only does it judge whether there exists a path but also gives a practical solution.
5. The worst-case complexity is exponential in the freedom degrees of the robotic arm, but seen from the angle of obstacle combinations, its time complexity is (XM^2), where M is the number of obstacles.

3. SIMULATION SYSTEM BASED ON DRM

We establish a simulation system for 3-joint manipulator Findpath planning based on DRM. Assume the three joints are $\{A_{i-1}A_i, l_i, r_i, \theta_i\}$ (i=1,2,3), where the arm $A_{i-1}A_i$ (i=2,3) rotates around the endpoint A_{i-1} in the same plane determined by $A_{i-2}A_{i-1}$ and $A_{i-1}A_i$ except that the arm A_0A_1 rotates around itself; A_0 is fixed; l_i is the length of $A_{i-1}A_i$; r_i is the radius of $A_{i-1}A_i$ and θ_i is the orientation of $A_{i-1}A_i$. Assume the environment is made up of obstacles such as pillars, cylinders. spheres and walls, which are not limited in number, size and location. See Fig 3.1.

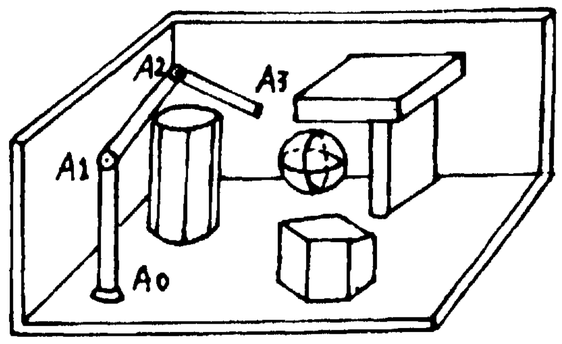


Fig 3.1 A robotic arm in an obstacle environment

CS: $\Theta_1\times\Theta_2\times\Theta_3$

RMG: $\{x, F(x),G(x,y)|x \in D(F),y \in F(x)\}$

DRM is applied recursively to C-Space in this way: $\Theta_1\rightarrow\Theta_2\times\Theta_3$ and $\Theta_2\rightarrow\Theta_3$, and CN is generalized to CN-Tree, whose every level corresponds to every degree of the robotic arm respectively. We must

realize the partition to enable every $(D_i, F_{j,k}(D_i), G_l(D_i, F_{j,k}(D_i)))$ is a connected block in RMG. It is obvious that the partition must be completed in two steps: first partition D into $\{D_i\}$ so that for every x in D_i $\mathbf{F(x)}$ has the same combination of connected branches and $F_j(x)$ has the same partition form, then compute {F,(D,)}; second partition $F_j(D_i)$ into $\{F_{j,k}(D_i)\}$ so that for every y in $F_{j,k}(D_i)$ $G(D_i, F_{j,k}(D_i))$ has the same combination of connected branches, then compute $\{G_l(D_i, F_{j,k}(D_i))\}$.

This is a framework of our simulation system:

* Input and organize obstacles with mathematical model based data structures and exclude the obstacles which are out of radius of l_2+l_3.
* Expand all obstacles by r (assume $r_1=r_2=r_3=r$), so the robotic arm is simplified as three line-segments jointing together. To maintain the independence of the algorithm, let us suppose that the expanded obstacle is a similar-shaped figure of the original one.
* Initialize 0-level Small-Node of CN-Tree;
* Partition on θ_1.
  * Project the environment vertically and obtain the topview. On the topview, compute the Supporting Lines [10] of every obstacle from the origin where the robotic arm is fixed, and arrange them in the angular order. Partition the whole possible angle range on Θ_1, $(\theta_{1,min}, \theta_{1,max})$ into angle intervals $\{(\alpha_i, \alpha_{i+1})\}$ with the sorted angle values.

After such partition, we can deduced that if we make a cross section at any angle value of (α_i, α_{i+1}), then on all of the cross sectioas there'll be the same number and the same shapes of cut obstacles, whereas on each cross section the sizes of obstacle sectional views may very and the distances between them may be different. See Fig 3.2 . Therefore the above partition is not enough for A_1A_2 and A_2A_3 to have the same connected branches on the cross sectioas made at every angle value in each interval (α_i, α_{i+1}). The cross sections at different angles of the same D_i may be different.

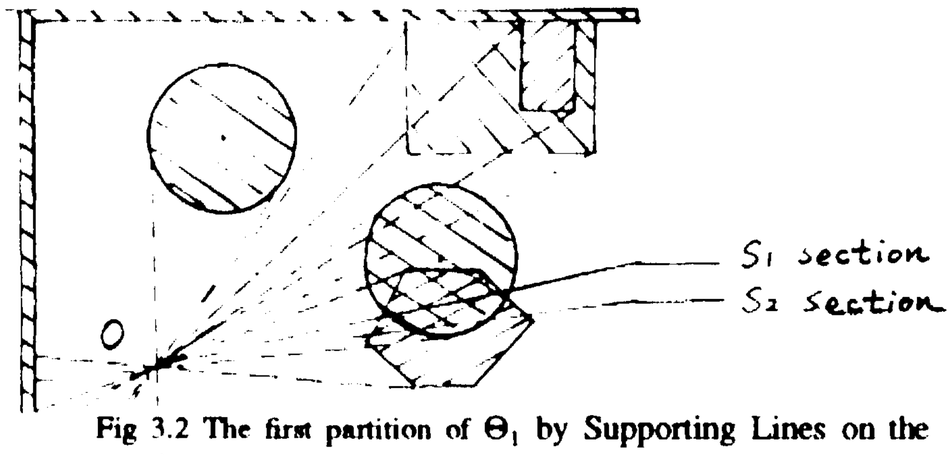


Fig 3.2 The first partition of Θ_1 by Supporting Lines on the topview

  * **Use l_2, l_2+l_3 and l_2-l_3 radius spheres to intersect with the obstacles. Project their intersection vertically and transfer them to polar-coordinate system, then select the maximum and minimum polar values and insert them to the above partition. See Fig 3.3.**

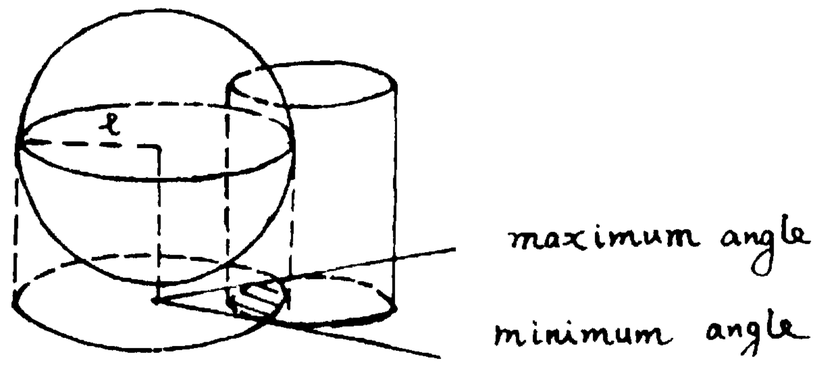


Fig 3.3 The maximum and minimum angles of the intersection of a l-radius sphere and an obstacle.

This is a more detail partition. If the Disappearance Curves don't exist at all then the partition will be enough for $A\ yA_2$ and $A\text{-}jA^{\wedge}$ to have the same connected branches in each partitioned angle interval on ©,. However, there are often Disappearance Curves when the obstacles are crowded together.

  * With the liner-cutting-detection technique, which means making cross sections at regular intervals within the partitioned interval, (α_i, α_{i+1}), detecting whether some DC emerges or vanishes, we can find the angle values on Θ_1 where the Disappearance Curves corresponding to A_1A_2 or A_2A_3 start or end. On Θ_1, the interval, (α_i, α_{i+1}), is so small after the former partition that the liner-cutting is very limited. Insert these angle values into the above partition. Now the partition on Θ_1 is at last completed.

Split the corresponding 0-level Small-Node to 1-level Big-Nodes on CN-Tree according to the complete partition. It is obvious that Big-Node levels maintain the adjacency information.

Because we are sure that every angle value belonging to one partitioned angle interval on Θ_1 , A_1A_2 and A_2A_3 will have the same connected branches, we make a cross section at an arbitrary value of every angle interval.

Compute the connected branches of A_1A_2 and expand the corresponding 1-level Big-Node to 1-level Small-Nodes on CN-Tree according to the computation

Partition each connected branch of A_XA_2 into sub-branches with the l_3-Growing Boundaries, Disappearance Curves corresponding to A_2A_3 and r-expanded obstacle boundaries so that for every angle value belonging to each sub-branch of A_1A_2, A_2A_3 has the same connected branches. See Fig 4.2.

Split the corresponding 1-level Small-Node to 2-level Big-Nodes on CN-Tree according to the partition on the cross section.

Compute the connected branches *oi AjA^* and expand the corresponding 2-level Big-Node to 2-level Small-Nodes on CN-Tree.

By now the CN-Tree is accomplished, which is also a 2-step tree. On the CN-Tree, each branch from the root node to the leaf node indicates a connected block in the C-Space of the robotic arm, or corresponds to a complete
connected branch.

* Link tiie nodes of the CN-Tree with arcs by applying theorem 2.2 to the CN-Tree recursively, so all connectivity information of the problem is stored in the CN-Tree.
* For a branch from the root to leaf on the CN-Tree is homologous with a leaf node on the CN-Tree, we deliver all connectivity information down to the leaf level, so we gain s simple graph, called Leaf-Graph.
* I ocate the initial and desired final state of the robotic arm to the corresponding leaf mxles on CN-Tree and search in the Leaf-Graph.

4. EXAMPLE

With one solution of our simulation system, we conclude this paper. The program is composed in PASCAL and performed on SUN 3/60 workstation. The CPU time of the following example is 1:17 seconds and we can still improve it by doing efforts.

See Fig 4.5, the robotic arm has three joints and the environment consists of a wall comer, a cylinder and two cuboids. **Three special states are given: S_1 , S_2 , S_3 , and they are all on the cross section shown in Fig 4.2. Because of the special behavior of A_0A_1 and the difference between the real robotic arm and the simplified three jointing line-segments, a state in Fig 4.3 can really corresponds to two real configurations of the robotic arm, e.g. S_0 and S_0'. If S_0 is $(\theta_1^{(0)}, \theta_2^{(0)}, \theta_3^{(0)})$ then S_0 will be $(\theta_1^{(0)}+\frac{\pi}{2},$**

$2\pi-\theta_2^{(0)}$, $2\pi-\theta_3^{(0)}$). So we have another three corresponding special states: S_1', S_2' and S_3'

In this case, the partition on 9, is relatively simple, see Fig 4.1. The CN-Tree is illustrated in Fig 4.4, where the lines of (lashes on the two Big-Node level represent the adjacency relations of D, and *FjK(D;)* respectively, the real lines on the Leaf-Node level represent the direct connectivity relations obtained from every cross section and the lines of dashes on the Leaf-Node level represent the indirect connectivity relations obtained by delivering connectivity information down. Finally all connectivity information is stored on the Leaf-Node level. The CN-Tree is symmetrical. The Leaf-Nodes where the states S1, S_2 and S_3 are located are also symmetrical with those where S_1', S_2' and S_3' are located, see Fig 4.4. This reflects the real world perfectly.

Searching in the Leaf-Graph, we can see that every two configurations of {S_1, S_2, S_3, S_1', S_2', S_3'} are connected. If we abbreviate a Leaf-Node (D_i, $F_{j,k}(D_i)$, $G_1(D_i, F_{jk}(D_i))$) to a four group (ijKl), then we can show a solution of $S_1 \rightarrow S_2 \rightarrow S_3$ by giving a series of the Leaf-Nodes: (3,2,1,1) → (3,2,2,1) → (3,2,3,1) → (4,1,1,1) → (5,2,1,1) → (4,2,2,1) → (33,2,2). Fig 4.5 is the hardcopies of the result displayed on the video-terminal of SUN 3/60 by selecting a arbitrary configuration in each of the nodes.

We adopt the Shortest-Way Algorithm of Graph Theory in our searching of the Leaf-Graph, then the solution is optimal in the sense of least sudden changes of states because a configuration can be turned into all configurations of the same Leaf-Node smoothly and continuously. This character is beneficial to the motion of a real robotic arm.

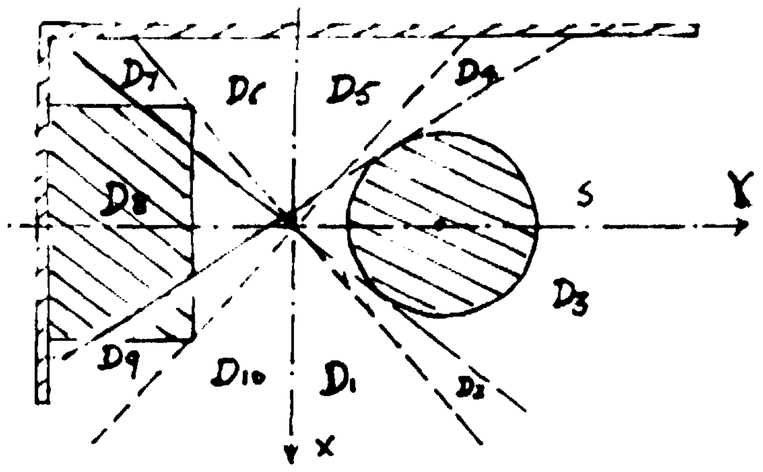


Fig 4.1 The partition on Θ_1 in the example is shown on the topview.

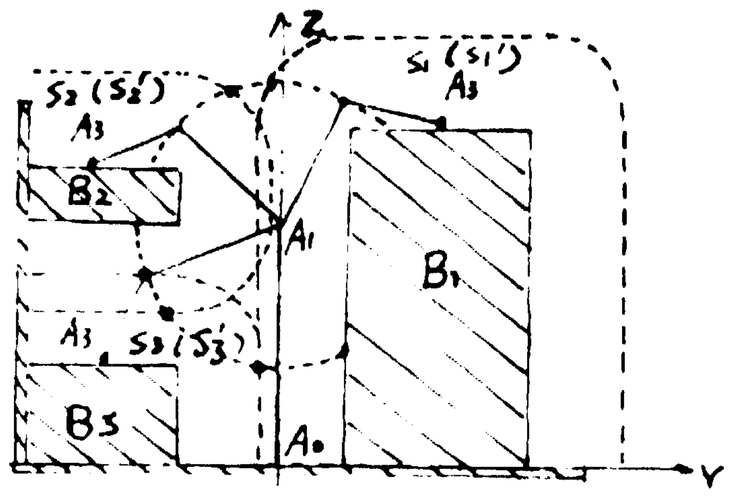


Fig 4.2 Analysis of a cross section y=0 from D_3 in the example: Assume the six special configurations (S_1, S_2, S_3, S_1', S_2', S_3') are on this cross section. $F_j(\theta_1)$ is partitioned by l_3-Growing Boundaries.

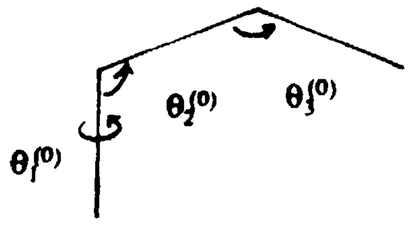


Fig 4.3 A state, S_0, is $(\theta_1^{(0)}, \theta_2^{(0)}, \theta_3^{(0)})$.

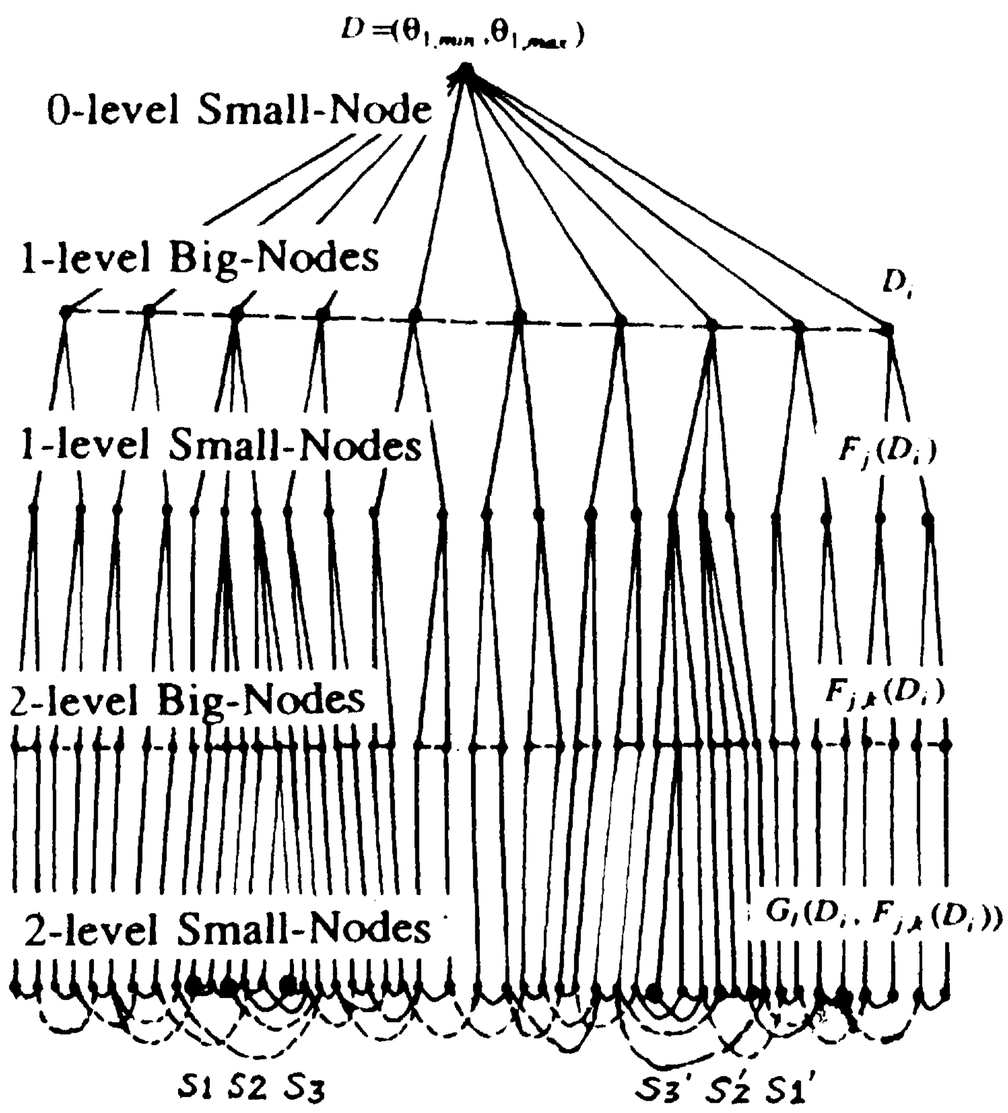


Fig 4.4 The CN-Tree and Leaf-Graph of the example

5. REFERENCE

[1] R.T. Chien, Ling Zhang, Bo Zhang, "Planning Collision-Free Paths For Robotic Arm Among Obstacles", IEEE Trans.PAMI., Vol.PAMI-6, No.I, Jan.,1984.

[2] Bo Zhang, Ling Zhang, "Planning Collision-Free Paths For 3-Dimensional Object With Rotation", Report on ACADEMIA SINICA (China) and C.N.R.(France) Robotics Workshop, Oct.,1985.

[3] Bo Zhang, Jianwei Zhang, Ling Zhang, "An Algorithm For Findpath With Rotation", Proc. of IEEE 1988 International Conference on SMC. Beijing. 795-798.

[4] Bo Zhang, Jianwei Zhang, Ling Zhang, Tian Zhang, "A Findpath Algorithm For Manipulator By Finite Division Of Configuration Space", Robotics and Manufacturing. Recent Trends in Research, Education and Application, M.Jamshidi(eds), ASME Press, New York, 1988, 99-106.

[5] T.Lozano-Pe'rez, "Spatial Planning: A Configuration Space Approach", IEEE Trans, on Comput., Vol.c-32, No.2, Feb.,1983.

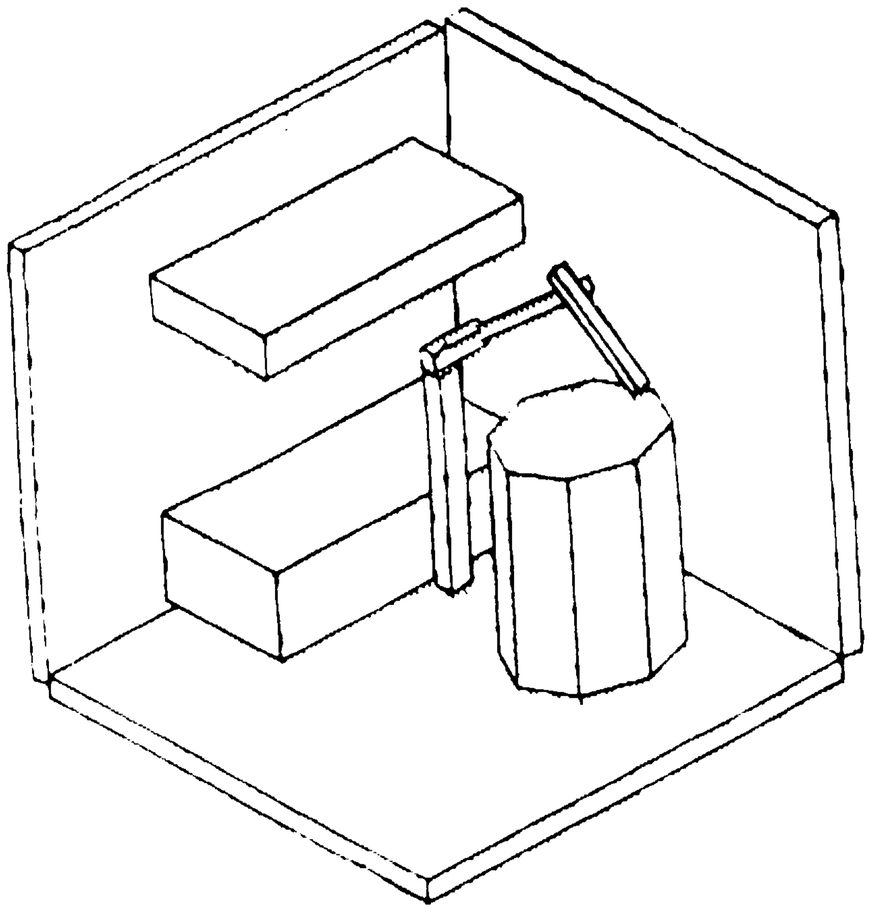
[6] R.A.Brooks, T.Lozano-Pe'rez, "A Subdivision Algorithm In Configuration Space for Findpath With Rotation", Proc. of 8th IJCAJ-83, pp.789-806.

[7] T.Lozano-Pe'rez, "A Simple Motion Planning Algorithm For General Robot Manipulators", IEEE Journal of Robotics and Automation, Vol.RA-3, No.3, June, 1987.

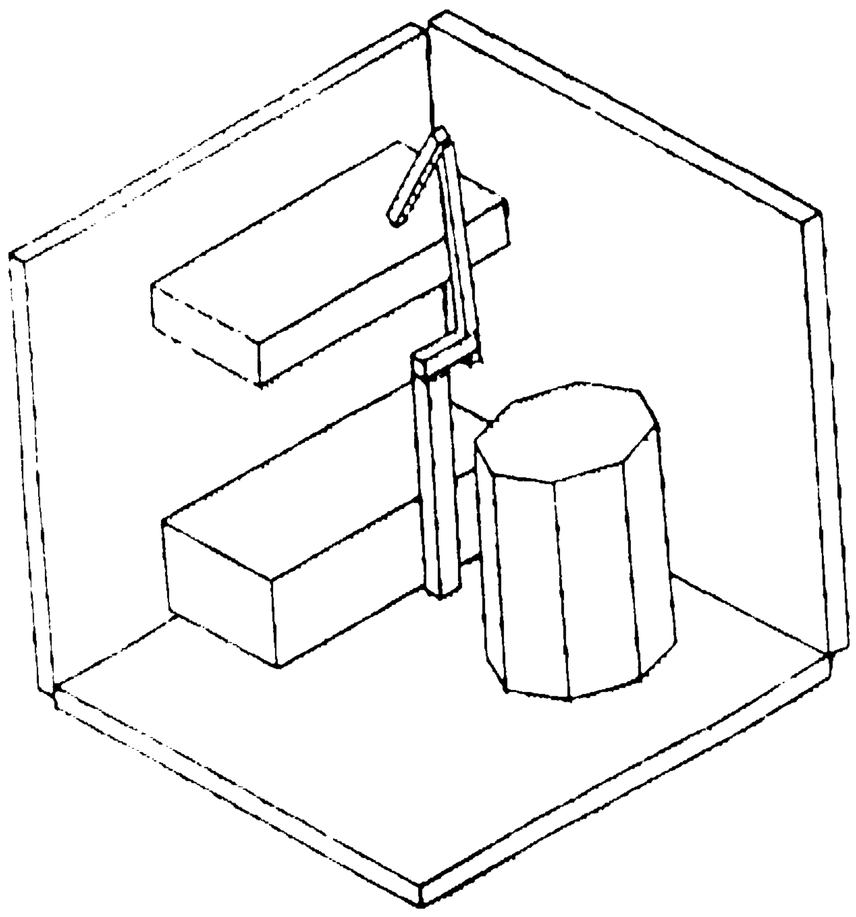
[8] J.T.Schwartz, M.Sharir, J.Hopcroft, "Planning Geometry And Complexity of Robot Motion", ABLEX PUBLISHING CORPORATION, Norwood,New Jersey, 1987.

[9] S.H.WhUesides, "Computational Geometry And Motion Planning", G.T.Toussaint (Editor), Elserier Science Publishers B.V.(North-Holland), 1985.

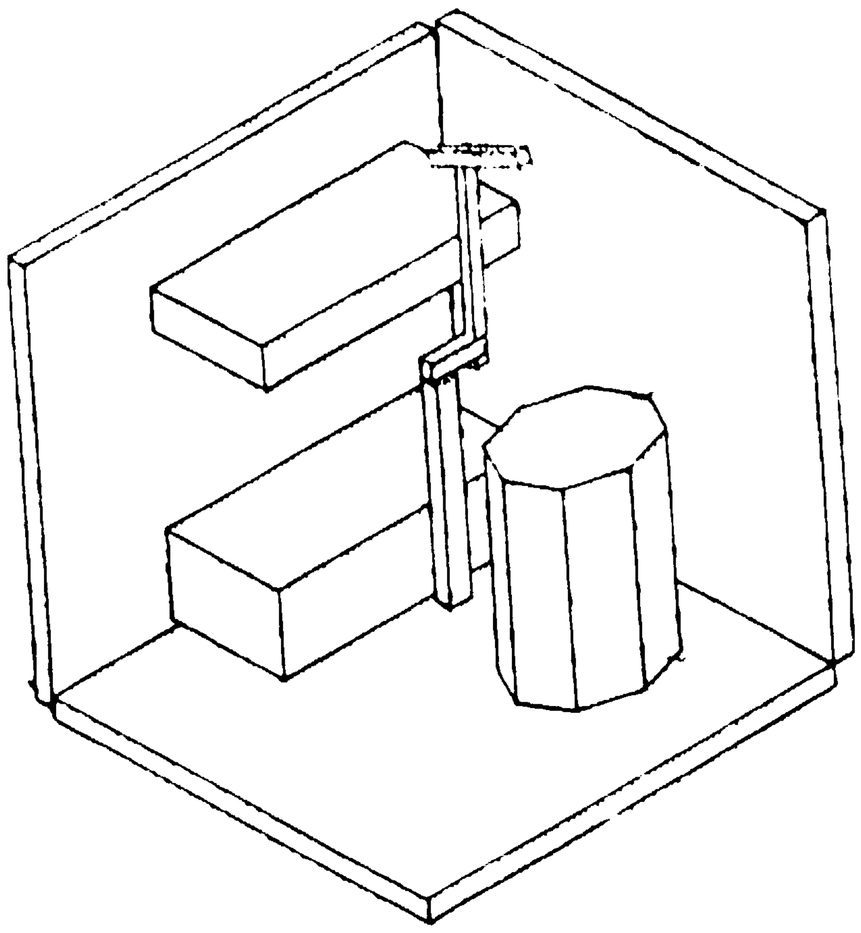
[10] F.P.Preparata, M.L.Shamos, S-Verley, "Computational Geometry-An Introduction". Texts and Mono-Graphs in Computer Science, 1985.



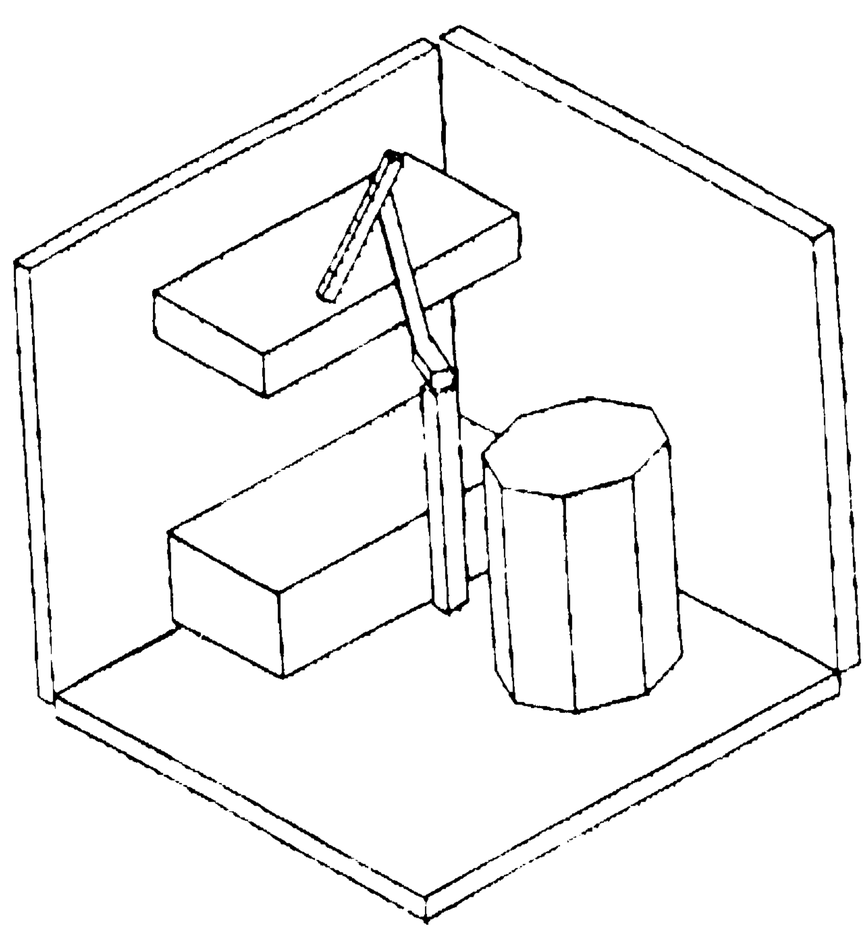
Step 1



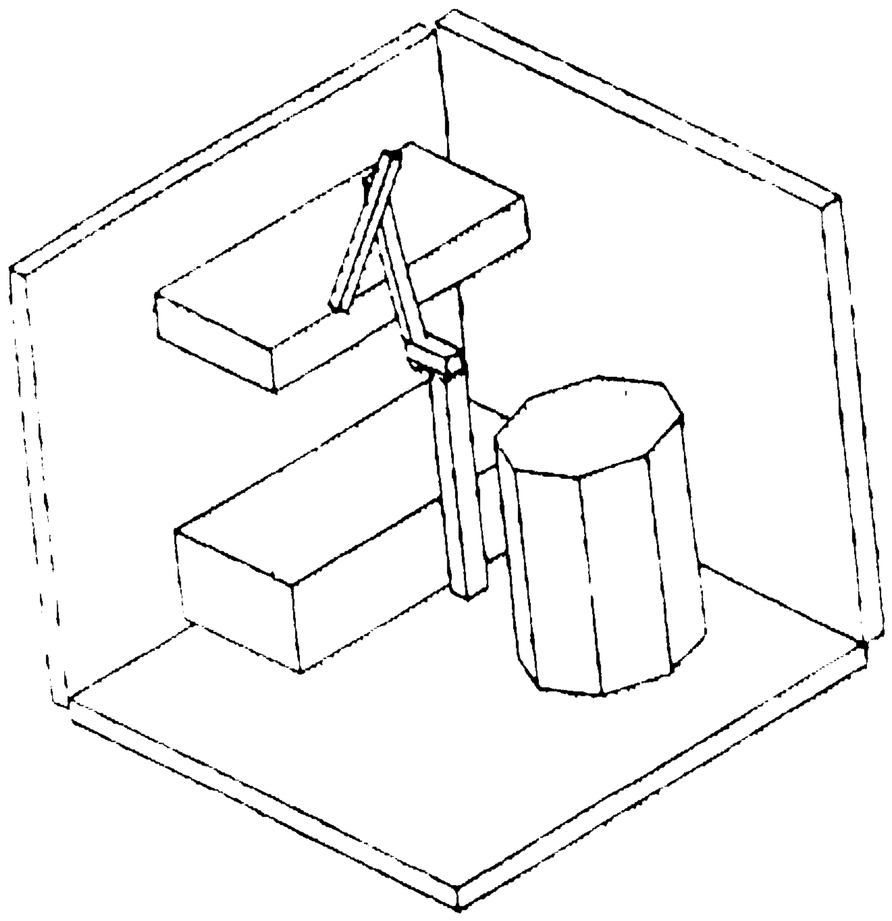
Step 3



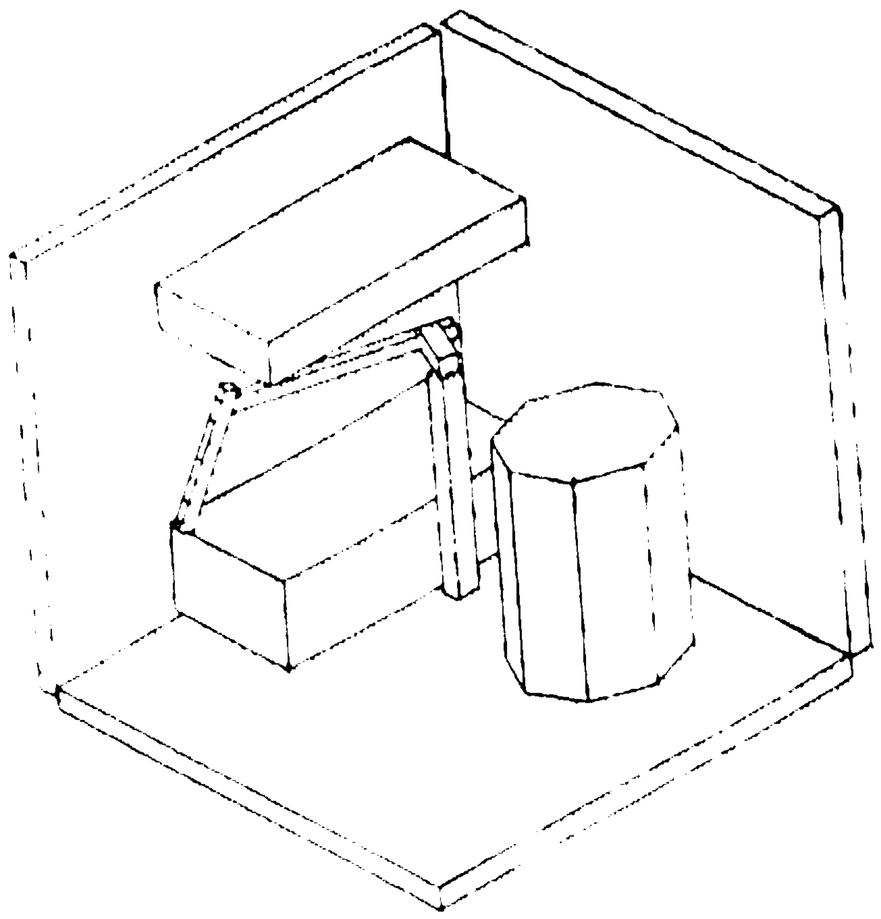
Step 2



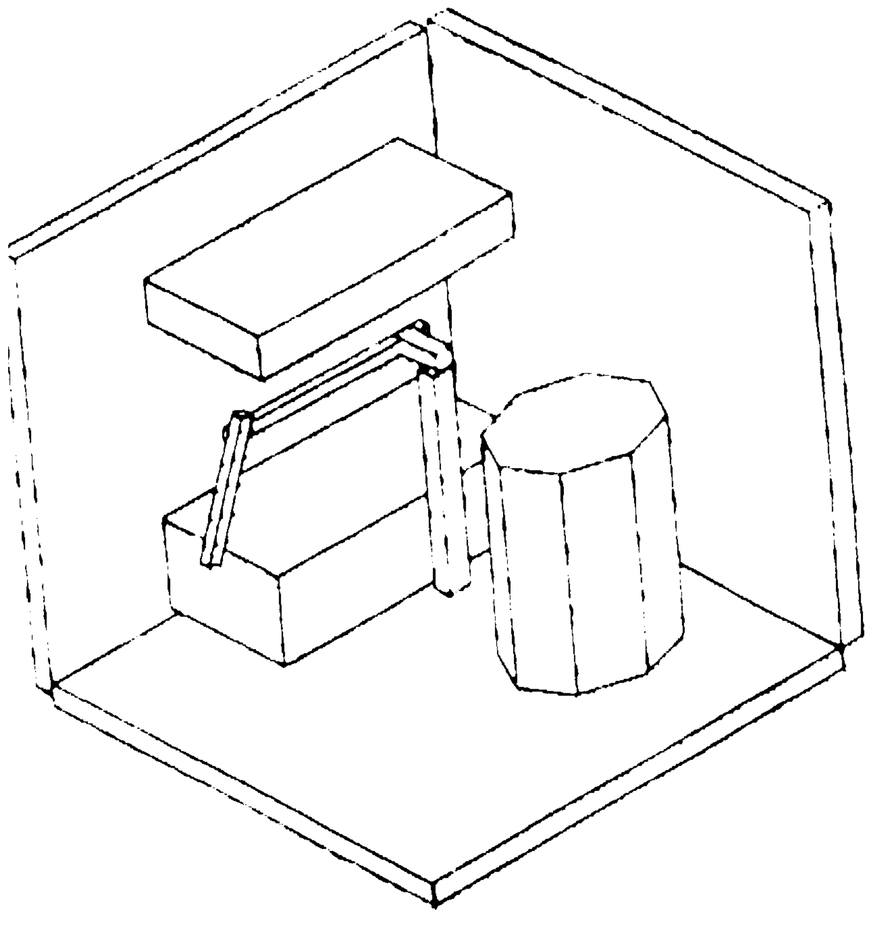
Step 4



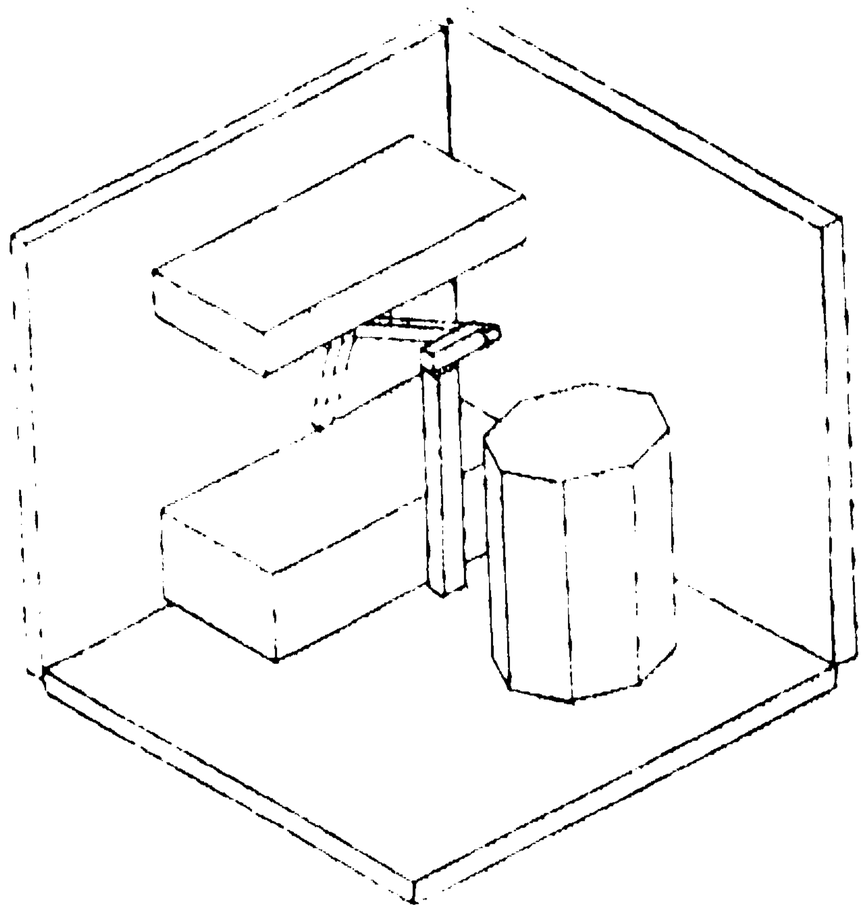
Step 5



Step 7



Step 6



Step 8

Fig 4.5 The solution to $S_1 \rightarrow S_2 \rightarrow S_3$ *is* displyed