
DETERMINING MANIPULATOR WORKSPACE BOUNDARIES USING THE MONTE CARLO METHOD AND LEAST SQUARES SEGMENTATION

David G. Alciatore

Department of Mechanical Engineering
Colorado State University
Fort Collins, Colorado

Chung-Ching D. Ng

Department of Mechanical Engineering
Colorado State University
Fort Collins, Colorado

ABSTRACT

Many investigators have developed methods for determining the workspace of a manipulator. One method presented by J. Rastegar and D. Perel uses the Monte Carlo method to generate a manipulator's workspace and its approximate boundary surfaces. Since it involves no inverse Jacobian calculation, problems dealing with singularity positions do not exist. However, only a graphical representation of the workspace is provided. The purpose of this work was to further develop the approach in order to determine an analytical description of a general 2D manipulator's workspace. The end result is a series of straight line and arc segments describing the workspace boundaries which were determined and segmented by least-squares line and arc fitting methods. Simple end-effector trajectory sweeping is also presented to aid in the segmentation.

1. INTRODUCTION

The workspace of a manipulator is defined as the set of points that can be reached by its end-effector. Analyzing workspace characteristics and shape provides a means to evaluate the efficiency and the kinematic performance of the manipulator mechanism. Many methods have been presented in the literature to determine a manipulator's workspace and its boundary surfaces, most of which can be categorized as analytical methods or numerical methods [1-15].

Analytical methods determine closed form descriptions of workspace boundary surfaces but these methods are usually complicated by nonlinear equations and matrix inversion involved in manipulator kinematics. Moreover, many analytical methods can only handle certain specific manipulators. For example, some methods are restricted to only revolute joint manipulators [1,2,3,8,10]. Numerical methods, on the other hand, are relatively simple and more flexible. However, they usually only determine approximate boundary surfaces and only represent them graphically [3,4,11,14].

Many approaches in the literature involve expressing and rearranging the manipulator kinematics equations [1-7].

Another common approach is geometric modeling. Chen and Gupta [10] utilized this powerful technique to determine manipulator workspaces and used Radial-Slice-Layering method and Apparent-Contour method in a CAD environment to find the workspace boundaries and represented them with bicubic spline curves. Another approach was proposed by Hansen, Gupta, and Kazerounian [8]. This method uses geometric sweeping to develop the manipulator workspace of n-R manipulators.

Other approaches are also presented in the literature [11-13]. On the whole, most of the approaches described are restricted to special manipulator types, and often the boundary surfaces are not represented analytically and are sometimes approximations.

Rastegar and Perel [14] introduced the Monte Carlo random sampling numerical method to generate the workspaces and the boundary surfaces of some simple manipulators using only forward kinematics. The method is relatively simple to apply. Unfortunately, the method only provides approximate graphical depictions and no analytical representations. The purpose of this paper is to extend the Monte Carlo approach to determine an analytical description of workspace boundary surfaces for general 2-D manipulators and to display them graphically. Such an analytical description is useful for accurate graphical display and for non-visual workspace reach and testing studies. The method developed is also extendible to 3-D manipulators.

2. IMPLEMENTATION

The 2D manipulator in Figure 1 will be used as an illustrative example. It has two links where joint 1 is a revolute joint and joint 2 is a prismatic joint. The position of the end-effector $P(x,y)$ is a function of the fixed link length a_1 and two variables: θ_1 and a_2 . Forward kinematics yield:

$$x = (a_1 + a_2) \cos \theta_1 \quad (1)$$

$$y = (a_1 + a_2) \sin \theta_1 \quad (2)$$

For this example, the joint limits are : $-45^\circ \leq \theta_1 \leq 45^\circ$ and $0 \leq a_2 \leq a_{2max}$. Also, $a_1=5$ and $a_{2max}=3$.

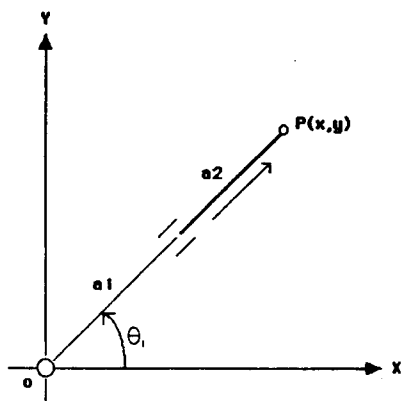


FIGURE 1 : RP MANIPULATOR EXAMPLE

(a) Generating approximate workspace

The Monte Carlo method of random sampling is applied to the joint space of the manipulator to approximate the workspace. For the example, the computer program picks values for the two variables θ_1 and a_2 randomly and generates 100,000 samples of the workspace. The values of θ_1 and a_2 are used to calculate the position of the end-effector P with forward kinematics and a dot is plotted at that position on the screen for each sample. Figure 2 shows the workspace for the first 10,000 points. A large number of points is required for getting good results in subsequent steps of the method.



FIGURE 2 : 10,000 MONTE CARLO POINTS

(b) Depicting boundary surfaces

To acquire the boundary surfaces, the workspace is tessellated into a square grid as illustrated in Figure 3. For each element having at least one Monte Carlo point, a 1 is stored in it and the average x and y values (i.e., the centroid) of the points within the element are calculated. A 0 is assigned to each element containing no points.

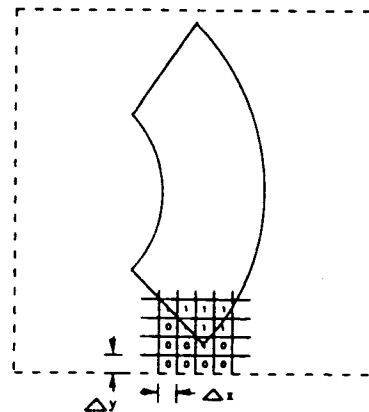


FIGURE 3 : DIGITIZATION OF THE WORKSPACE

Elements are inspected one by one from left to right and from bottom to top. An element is flagged as a boundary element (i.e., turned on) only if the following two conditions are satisfied: 1) a 1 is stored; 2) at least one of its eight neighbor grids has a 0 stored (see Figure 4). Elements which fail to fulfill either of these conditions are not on the workspace boundary and are turned off.

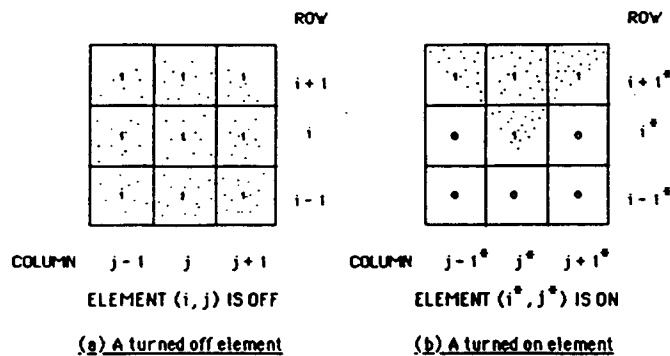


FIGURE 4 : BOUNDARY ELEMENT NEIGHBOR TEST

For those elements that are on, a dot is plotted at each corresponding centroid previously calculated. At this point the workspace boundaries are represented by a set of approximate boundary points (see Figure 5).



FIGURE 5 : APPROXIMATE BOUNDARY POINTS

(c) Searching critical points

For any arbitrary 2D manipulator which has prismatic joints and revolute joints, the workspace boundaries can always be analytically described by connected straight lines and circular arcs. For the example, the boundary contains two straight lines and two circular arcs (see Figure 5). The most important and most difficult step is to segment the boundary locating the critical points A, B, C, and D which separate these lines and arcs. Once they are found, the least-squares method is applied to segment lines and arcs. The algorithm developed to solve this problem is presented below step by step:

1. Determine Pure Joint Limit Critical Points.

Determine all the positions of the end-effector at all joint limit combinations, i.e., all combinations of minimum and maximum joint values. For the example, there are four combinations: i) $\min \theta_1, \min a_2$; ii) $\min \theta_1, \max a_2$; iii) $\max \theta_1, \min a_2$; and iv) $\max \theta_1, \max a_2$. They represent critical points B, C, A, and D respectively.

2. Eliminate Non-critical Points Positions.

Positions calculated in step 1 are critical points if they lie on the boundary. Those within the boundary are not. To check whether one of these positions is a critical point or not, the minimum distance between this location and the set of boundary points is determined. If this value is small, the point is considered on the workspace boundary. By doing this, non-critical points are removed. Since A, B, C, and D are all critical points, none of them are discarded.

It is possible that critical points are located at positions having partial joint limit combinations where only one or several of the manipulator joints are at either a minimum or maximum joint value while the others are not (points 2, 3, 5, 6, 7 and 8 in example 3 are such points). The method for determining partial joint limit critical points is described in step 4.

3. Determine the Connectivity Type Between Pure Joint Limit Critical Points.

Between the critical points in example 1 there are N boundary points (x_i, y_i) where $i = 1, 2, \dots, N$. The least-squares method is used to determine whether the boundary segment is a line or an arc. A perpendicular least-squares line fit [16] is used to determine a line segment error:

$$LSE = \sum_{i=1}^N d_i^2 \quad (3)$$

A similar least-squares arc fit [17] is used to determine an arc segment error:

$$ASE = \sum_{i=1}^N [(x_o - x_i)^2 + (y_o - y_i)^2 - R^2]^2 \quad (4)$$

If LSE is smaller than a certain tolerance (TOLL), the points are assumed to be represented by a line (between critical points B and C in example 1). By the same token, if ASE is less than another tolerance (TOLA), a circular arc is then used to represent the points (between critical points A and B in

example 1). Finally, if neither error is within the tolerance, it is assumed that there exists at least one partial joint limit critical point between them. TOLL and TOLA increase as the number of points increases.

4. Determine Partial Joint Limit Critical Points.

Consider Figure 6 as an illustrative example and suppose that pure joint limit critical points C1, C2, C3 and C4 are known. To locate the first two unknown critical points C1-2₁ and C1-2₂ between C1 and C2, the bi-section method is used. First of all, the techniques introduced in steps 3 are applied to the points between C1 and C2. If the two conditions in step 3 are not fulfilled, the points are divided into two sub-groups and the same techniques are applied to the first sub-group. This procedure is repeated recursively until either $LSE < TOLL$ or $ASE < TOLA$ occurs.

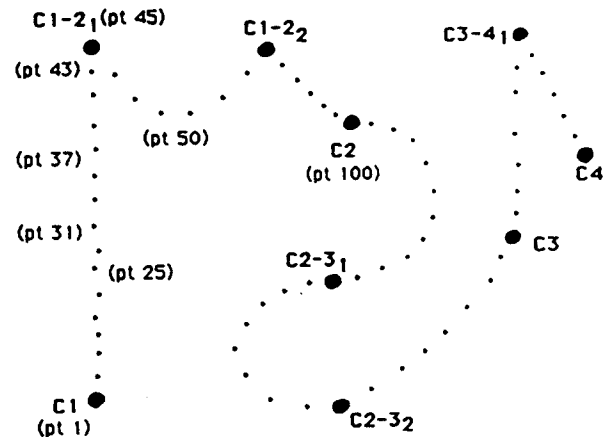


FIGURE 6: WORKSPACE BOUNDARY WITH MULTIPLE CRITICAL POINTS

Assume that C1 is point 1 and C2 is point 100. Suppose that $LSE < TOLL$ is true when the first sub-group of points contains point 1 through point 25 (step 3 has been repeated twice). The result implies that a line is assumed to represent point 1 through point x where x is somewhere between 25 and 50. Step 3 is applied to point 1 through point 37 ($\cong (25+50)/2$). If LSE is larger than TOLL, bisect from 37 to 31 ($\cong (25+37)/2$). If LSE is smaller than TOLL, bisect from 37 to 43 ($\cong (37+50)/2$). The process stops when an answer converges.

Once an extra critical point has been found, in this case C1-2₁ which is point 45, the algorithm is repeated again for point 45 through point 100. When the other critical point C1-2₂ between C1 and C2 is also located, two other known critical points are picked every time (start with C2 and C3) and the whole process repeats several times to look for C2-3₁, C2-3₂, C3-4₁, etc.

(d) Presenting analytical description

For a general 2D manipulator the analytical description of the workspace boundaries consists of lists of connected line and arc segments. The output file for example 1 is:

	x_s	y_s	x_e	y_e	line/arc	x_o	y_o	R
AB	3.53	3.54	3.54	-3.53	ARC	0.061	-0.004	5.04
BC	3.54	-3.53	5.65	-5.66	LINE			
CD	5.65	-5.66	5.66	5.66	ARC	0.057	0.011	7.92
DA	5.66	5.66	3.53	3.54	LINE			

where:

$x_s, y_s = x, y$ coordinates of the starting position;
 $x_e, y_e = x, y$ coordinates of the ending position;
line/arc = connectivity type between (x_s, y_s) and (x_e, y_e) ;
 $x_o, y_o = x, y$ coordinates of the circular arc center ; and
R = radius of the circular arc.

3. ADDITIONAL EXAMPLES

Two additional examples are shown in this section. Each example includes: a) corresponding input file, b) approximate workspace, c) approximate boundary surfaces with critical points, d) segmented lines and circular arcs, and e) corresponding output file.

The following data were input for each example:

- 1) number of Monte Carlo points to be generated (N_m).
- 2) number of grid elements (N_g).
- 3) number of joints (N).
- 4) joint type for each joint (R-revolute or P-prismatic).
- 5) link twist (α), link length (a), link offset (d), and joint angle (θ) for each joint.

Refer to [18] about the forward kinematics system used in this paper.

Example 2 - RR MANIPULATOR :

$N_m = 100000, N_g = 2500, N = 2$, joints : R R,

i	a_{i-1}	a_i	min d_i	max d_i	min θ_i	max θ_i
1	0	0	0	0	0	360
2	0	8	0	0	0	360
3	0	5	0	0		

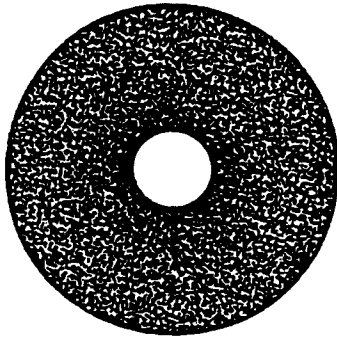


FIGURE 7 : EXAMPLE 2 APPROXIMATE WORKSPACE

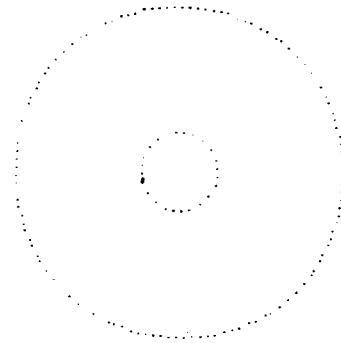


FIGURE 8 : EXAMPLE 2 APPROXIMATE BOUNDARIES WITH CRITICAL POINTS

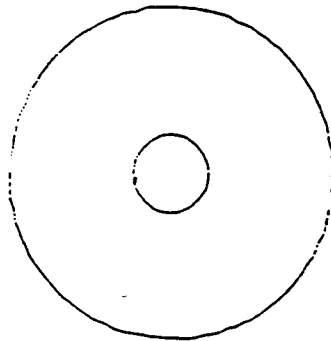


FIGURE 9 : EXAMPLE 2 SEGMENTED LINES AND ARCS

x_s	y_s	x_e	y_e	line/arc	x_o	y_o	R
13	0	13	0	ARC	-0.0023	.00026	12.91
-2.92	-0.96	-2.92	-0.96	ARC	-.0017	.0035	3.05

Example 3 - PR MANIPULATOR :

$N_m = 100000, N_g = 2500, N = 2$, joints : P R,

i	a_{i-1}	a_i	min d_i	max d_i	min θ_i	max θ_i
1	-90	0	0	6	0	0
2	90	0	0	0	0	360
3	90	0	2	2		



FIGURE 10 : EXAMPLE 3 APPROXIMATE WORKSPACE

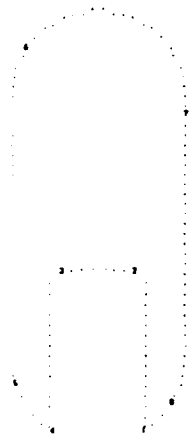


FIGURE 11 : EXAMPLE 3 APPROXIMATE BOUNDARIES WITH CRITICAL POINTS

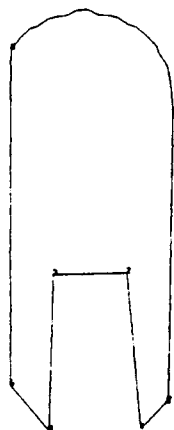


FIGURE 12 : EXAMPLE 3 SEGMENTED LINES AND ARCS

	x_s	y_s	x_e	y_e	line/arc	x_o	y_o	R
12	1.02	-1.70	0.82	1.91	LINE			
23	0.82	1.91	-0.81	1.91	LINE			
34	-0.82	1.91	-1.02	-1.70	LINE			
45	-1.02	-1.70	-1.71	-0.87	LINE			
56	-1.87	-0.61	-1.64	7.10	LINE			
67	-1.64	7.10	1.96	5.57	ARC	-0.0066	5.97	1.97
78	1.96	5.57	1.66	-1.04	LINE			
81	1.66	-1.04	1.02	-1.70	LINE			

4. SUMMARY

An algorithm was developed in this paper to determine the analytical representations of general 2-D manipulator workspace boundary surfaces and display them graphically. The algorithm works well for most manipulators but it has some imperfections. The critical points illustrated in Figure 11 are not the real critical points positions, they are only approximate.

There are two main problems in the algorithm that cause errors in locating critical points. The first problem is related to applying the Monte Carlo method. One disadvantage in using the Monte Carlo method is that most of the points it yields are within the workspace boundaries instead of on the boundary surfaces. Therefore, the digitization process do not depict exact boundaries but only approximate boundaries. The second problem is a result of limitations of the bisection method. The method is not perfect because the distinction between line and arc becomes unclear at small scales.

In general, trying to segment a collection of boundary points into connected lines and arcs is a difficult problem which does not have a general and robust solution. One solution for finding the exact critical point locations is to combine the Monte Carlo method with an end-effector swept-boundary-intersection method. This method involves successively sweeping the end-effector point through each joint range of motion starting at the last joint. Only the boundaries (lines and arcs) of the swept area are generated and stored for each joint motion sweep. General area sweeping [19,20] can result in the final workspace area, but this is a more difficult problem. The swept-boundary-intersection method proposed here does not require area determination; only the area boundaries have to be generated. The manipulator in example 3 will be used as an illustrative example to explain this sweeping method.

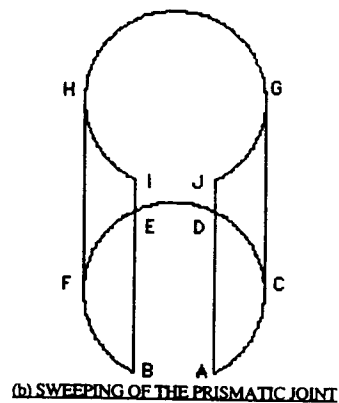
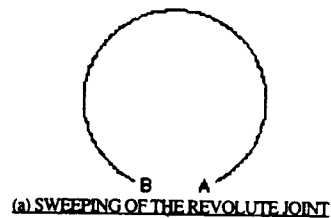


FIGURE 13 : SWEEPING OF A PR MANIPULATOR

First of all, the end-effector point is swept through the revolute joint. The resulting end-effector point trajectory is a circular arc where A and B are the end points (see Figure 13a). Next the first link is swept along the prismatic joint and the original end-effector trajectory is swept linearly (see Figure 13b). Then this boundary arc is swept through the prismatic motion resulting in two arcs (IJ and BA) and four lines (HF, IB, JA, GC). All of the intersection points between these swept line and arc boundary segments are candidate critical points (A, B, C, D, E, F, G, H, I and J).

If any of the potential critical points is close to the approximate boundary surfaces previously extracted from sampling (see Figure 11), it is a true critical point (A, B, C, D, E, F, G and H); otherwise it is not and can be ignored (I and J). By knowing the real critical point positions and using the least-squares line and arc fitting methods (or alternatively using the swept boundary information), the workspace boundary can be segmented and analytically described. A complete robust general solution for planar manipulators is achieved.

Although this paper only dealt with 2D manipulators, the algorithm is extendible to 3D manipulators. The squared grid elements discussed in the 2D environment would become cubed volume elements in the 3D environment. Moreover, critical points in 2D would be represented by both critical points and edges in 3D. Finally, the segmented lines and arcs would be replaced by plane, spherical, and cylindrical segments.

5. REFERENCES

- [1] Tsai, Y. C., Soni, A. H., "An Algorithm for the Workspace of a General n-R Robot", ASME Journal of Mechanisms, Transmissions, and Automation in Design, March 1983, Vol. 105, pp. 52-57.
- [2] Yang, D. C. H., Lee, T. W., "On the Workspace of Mechanical Manipulators", ASME Journal of Mechanisms, Transmissions, and Automation in Design, March 1983, Vol. 105, pp. 62-69.
- [3] Lee, T. W., Yang, D. C. H., "On the Evaluation of Manipulator Workspace", ASME Journal of Mechanisms, Transmissions, and Automation in Design, March 1983, Vol. 105, pp. 70-77.
- [4] Kumar, A., Patel, M. S., "Mapping the Manipulator Workspace Using Interactive Computer Graphics", The International Journal of Robotics Research, Summer 1986, Vol. 5, No. 2, pp. 122-130.
- [5] Bacquet, M., "Computing of Robot's Workspace", Proceedings of 1985 International Conference on Advanced Robotics, pp. 289-294.
- [6] Rastegar, J., Deravi, P., "Methods to Determine Workspace, its Subspaces with Different Numbers of Configurations and All the Possible Configurations of a Manipulator", Mechanism and Machine Theory, 1987, Vol. 22, No. 4, pp. 343-350.
- [7] Rastegar, J., Deravi, P., "The Effect of Joint Motion Constraints on the Workspace and Number of Configurations of Manipulators", Mechanism and Machine Theory, 1987, Vol. 22, No. 5, pp. 401-409.
- [8] Hansen, J. A., Gupta, K. C., Kazerounian, S. M. K., "Generation and Evaluation of the Workspace of a Manipulator", The International Journal of Robotics Research, Fall 1983, Vol. 2, No. 3, pp. 22-31.
- [9] Kohli, D., Spanos, J., "Workspace Analysis of Mechanical Manipulators Using Polynomial Discriminants", ASME paper No. 84-DET-121, pp. 1-7.
- [10] Chen, X., Gupta, K. C., "Geometric Modeling and Visualization of Manipulator Workplace", ASME Computers in Engineering, 1991, Vol. 1, pp. 469-474.
- [11] Kumar, A., Waldron, K. J., "The Workspace of a Mechanical Manipulator", ASME Journal of Mechanical Design, July 1981, Vol. 103, pp. 665-672.
- [12] Selfridge, R. G., "The Reachable Workarea of a Manipulator", Mechanism and Machine Theory, 1993, Vol. 18, No. 2, pp. 131-137.
- [13] Guo-dong Li, Ning-xin Chen, Qi-xian Zhang, "Analysis of Translational Singular Surfaces and Boundary Surfaces of Workspace of an Arbitrary Manipulator", ASME Trends and Developments in Mechanisms, Machines, and Robotics, 1988, Vol. 3, pp. 275-282.
- [14] Rastegar, J., Perel, D., "Generation of Manipulator Workspace Boundary Geometry Using the Monte Carlo Method and Interactive Computer Graphics", ASME Trends and Developments in Mechanisms, Machines, and Robotics, 1988, Vol. 3, pp. 299-305.
- [15] Rastegar, J., Fardanesh, B., "Manipulator Workspace Analysis Using the Monte Carlo Method", Mechanism and Machine Theory, 1990, Vol. 25, No. 2, pp. 233-239.
- [16] Alciatore, D., Miranda, R., "The 'Best' Least-Squares Line Fit", Submitted for publication in Graphics GEMF V, edited by James Arvo, Academic Press, 1994.
- [17] Moura, L., Kitney, R., "A direct method for least-squares circle fitting", Computer Physics Communications 64 (1991) 57-63.
- [18] Craig, J. J., Introduction to Robotics Mechanics and Control, Second Edition, Addison-Wesley, 1989.
- [19] Martin, R. R., Stephenson, P. C., "Sweeping of three-dimensional objects", Computer-Aided Design, May 1990, Vol. 22, No. 4, pp. 223-234.
- [20] Blackmore, D., Leu, M. C., "Analysis of Swept Volume via Lie Groups and Differential Equations", The International Journal of Robotics Research, December 1992, Vol. 11, No. 6, pp. 516-537.