Proceedings of the 2002 IEEE
International Conference on Robotics & Automation
Washington, DC • May 2002

Using Manipulability to Bias Sampling During the Construction of Probabilistic Roadmaps

Peter Leven, Seth Hutchinson
p.leven@computer.org, seth@uiuc.edu
University of Illinois at Urbana-Champaign
Urbana, IL, USA

Abstract

Probabilistic roadmaps (PRMs) are a popular representation used by many current path planners. Construction of a PRM requires the ability to generate a set of random samples from the robot's configuration space, and much recent research has concentrated on new methods to do this.

In this paper, we present a sampling scheme that is based on the manipulability measure associated with a robot arm. Intuitively, manipulability characterizes the arm's freedom of motion for a given configuration. Thus, our approach is to sample densely those regions of the configuration space in which manipulability is low (and therefore the robot has less dexterity), while sampling more sparsely those regions in which the manipulability is high.

We have implemented our approach, and performed extensive evaluations using prototypical problems from the path planning literature. Our results show this new sampling scheme to be quite effective in generating PRMs that can solve a large range of path planning problems.

1 Introduction

Probabilistic roadmaps (PRMs) were introduced in the early nineties as a representation useful for planning collision-free paths for robots with many degrees of freedom [11, 16]. PRM path planners use a two-stage approach. During a preprocessing stage, the planner generates a set of nodes that correspond to random configurations in the configuration space, connects these nodes using a local path planner to form a roadmap, and, if necessary, uses a subsequent sampling stage to enhance the roadmap. During a second, on-line stage, planning is reduced to query processing, in which the initial and final configurations are connected to the roadmap, and the augmented roadmap is searched for a feasible path.

The method used for generating the random configurations lies at the heart of any PRM planner. For this reason, numerous sampling schemes have been proposed in recent years. We review many of these methods below, in Section 2.

In this paper, we present a new method for biasing the sampling during the node generation stage used to build a PRM. Our method is based on *manipulability* [18], an intrinsic property of robot arms, which measures an arm's freedom to move in all directions. Our rationale for this approach is that in regions of the configuration space where manipulability is high, the robot has great dexterity, and therefore relatively fewer samples should be required in these areas. Conversely, regions in which the manipulability is low tend to be near (or to include) singular configurations of the arm, where the range of possible motions is reduced; therefore such regions should be sampled more densely. Our choice to use such an intrinsic property, rather than to drive sampling based on the geometry of the obstacle region of the workspace is motivated by our previous work in generating representations that can be used for path planning in changing environments [15, 14]. We describe our new approach in Section 3.

We have done extensive comparisons between our new approach and the original approach described in [9]. We present these results in Section 4, and in Section 5 we briefly present the conclusions that can be drawn from our results.

2 Related Work

The simplest way to generate sample configurations is to sample the configuration space uniformly at random, discarding the samples that lead to a collision. This technique makes no assumptions about the distributions of the obstacles and is relatively easy to analyze [10]. Unfortunately, the number of samples this technique places in any particular region of $\mathcal{C}_{free}$ (the set of collision-free configurations) is proportional to its volume; therefore, uniform sampling is unlikely to place samples in narrow passages. Most recent PRM sampling schemes have been developed to address this problem.

0-7803-7272-7/02/$17.00 © 2002 IEEE

One means to address the problem of few samples in narrow passages in $\mathcal{C}_{free}$ is to add samples in regions where the roadmap has few nodes. More samples near these nodes can be taken by randomly bouncing off obstacles: choose a random direction, travel in the direction until an obstacle is encountered, choose another direction, and continue until the path length reaches some threshold [7, 12].

Another technique for sampling $\mathcal{C}_{free}$ is to concentrate the samples near the surfaces of the obstacles in configuration space. One such approach is to locate samples on the surfaces of the obstacles themselves. This approach works by taking an arbitrary sample in collision and then searching for the boundary of the collision region of $\mathcal{C}$ (the configuration space) on rays directed away from the collision point, uniformly distributed on a hypersphere [3]. The hope is to locate the center point close to the center of the obstacle region such that the samples are uniformly distributed over the surface of the obstacle. A modified version of this approach also adds samples near the surfaces, in some cases building shells of samples around the $\mathcal{C}$-obstacles [2]. Another approach for concentrating the samples near the obstacle surface is to generate pairs of points, one uniform at random in $\mathcal{C}$ and the other a small distance away (with the distance guided by sampling a Gaussian distribution) [4]. In this case, the collision-free sample of the pair is added to the roadmap only if the second sample is in collision.

There are other techniques that use the geometry of the obstacles to define sample points. One such approach that works well for rigid-body robots in 2D environments is to use the geometry of the obstacles and the robot in the workspace to define the sample nodes in $\mathcal{C}_{free}$ [16]. In this case, the axis of the robot is placed parallel to obstacle surfaces with the robot a small distance away from the surface (a similar position is defined for vertices). Another geometric approach is to generate samples along the medial axis, either in $\mathcal{C}_{free}$ or in the workspace. To generate samples using the medial axis in the workspace, the idea is to first compute the medial axis of the workspace, and then take random configurations and move the robot from those configurations until some subset of reference points defined on the robot lie on (or as close as possible to) the medial axis [5, 6]. To generate samples in the medial axis in $\mathcal{C}_{free}$, the idea is to take random configurations and transform them to the medial axis [17]. Each random configuration falls into one of three cases. In the first case, the sample is on the medial axis and nothing further need be done. Second, the sample could be in $\mathcal{C}_{free}$ but not on the medial axis, in which case the point is translated away from the nearest obstacle until it is equidistant from two obstacles. Third, the sample could be in an obstacle region. In this case, the configuration is translated to the nearest obstacle surface, and then to the medial axis.

Some techniques are designed in particular for single-query path planning. One of these involves choosing a node at random from the current roadmap, generating samples around that node, and adding some of these new samples to the roadmap [8]. In this case, the node selection is biased towards the nodes with fewer neighbors, and new samples that have too many neighbors in the roadmap are rejected. Another technique in a similar vein is to generate a sample at random of $\mathcal{C}$, find the node nearest to it in the roadmap, and generate a new node in the roadmap by moving toward the random sample [13]. This use of the random sample should bias the tree to explore $\mathcal{C}_{free}$ more rapidly. The Ariadne's Clew algorithm also falls in this category, though it uses the endpoints of Manhattan paths as its samples [1]. Another approach for single-query path planning used a vision sensor to detect the free portions of the workspace, and a sampling algorithm that placed new nodes to explore further the free workspace [19].

3 Manipulability-based sampling

We have developed a new importance sampling approach that is based on the manipulability measure associated with the manipulator Jacobian matrix [18]. The basic idea for using the manipulability as a bias for sampling is the following. In regions of the configuration space where manipulability is high, the robot has great dexterity, and therefore relatively fewer samples should be required in these areas. Regions of the configuration space where manipulability is low tend to be near (or to include) singular configurations of the arm. Near singularities, the range of possible motions is reduced, and therefore such regions should be sampled more densely.

Let $J(\mathrm{q})$ denote the manipulator Jacobian matrix (i.e., the matrix that relates velocities of the end effector to joint velocities). For a redundant arm (i.e., an arm with more than six joints for a 3D workspace, and an arm with more than three joints for a 2D workspace) the manipulability in configuration q is given by

$$\omega(\mathrm{q}) = \sqrt{\det J(\mathrm{q}) J^T(\mathrm{q})}. \qquad (1)$$

Consider the robot shown in Figure 1 as an example. The manipulability for this robot is $\omega = l_1 l_2 |\sin\theta_2|$, where l_1 and l_2 are the lengths of the two links. The configuration shown in Figure 1 corresponds to one of the configurations at which the manipulability is

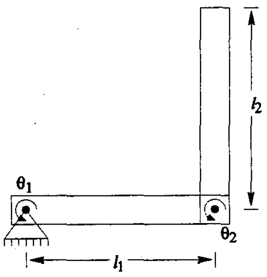

Figure 1: Planar robot example.

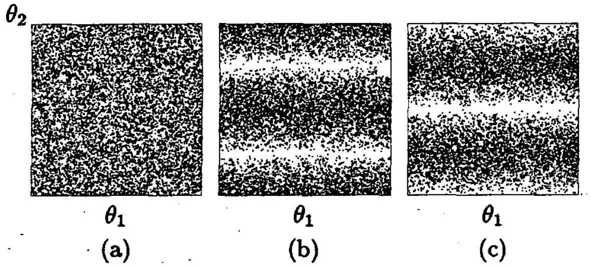

Figure 2: Sample distributions for a two-joint planar robot: (a) uniform, (b) higher density in regions of low manipulability, (c) higher density in regions of high manipulability.

highest for this robot (the other configuration has the second link pointing down). Notice that for this robot the manipulability does not depend on the position of the first joint.

Three different sample distributions for the robot are shown in Figure 2. As shown in the figure, concentrating the sampling in regions of low manipulability results in more samples near $\theta_2 = -\pi$, 0, and π at the bottom, middle, and top of the views of configuration space, respectively; sampling in regions of high manipulability results in more samples near $\theta_2 = -\pi/2$ and $\pi/2$. Normalizing for the link lengths, the average manipulability of the samples for the uniform distribution is 0.65, the average for low manipulability is 0.49, and the average for high manipulability is 0.82.

In order to bias sampling based on manipulability, we use an approximation of the cumulative density function (CDF) for manipulability. If we treat manipulability as a random quantity, denoted by the random variable Ω, with probability density function p_Ω, the CDF is given by

$$P_\Omega(\omega) = \int_0^\omega p_\Omega(t)dt.$$

We compute a discrete representation of P_Ω as follows. First, we create a discrete approximation to p_Ω. This is done by sampling the configuration space of the robot uniformly at random and computing the manipulability for each sample configuration. We exclude from this computation any configuration in which the robot collides with itself. We then create a histogram of the manipulability values that have been computed. We normalize the number in each bucket of the histogram, and create the approximation to P_Ω from these normalized values. The bucket size of the histogram and the number of samples to take are parameters.

We have adopted a rejection–based approach for using P_Ω to bias the sampling of the configuration. For each sample, we use the following procedure. First, a candidate sample, q_c is generated using uniform random sampling of the configuration space. If q_c is a self-collision configuration, it is rejected. If q_c is not rejected, we compute the manipulability $\omega(\mathrm{q}_c)$. We reject q_c with probability $P_\Omega(\mathrm{q}_c)$. This approach was used to generate the sample distribution in Figure 2(b). In Figure 2(c), we use $P_\Omega(\mathrm{q}_c)$ as the probability of acceptance.

One shortcoming of the manipulability measure for our purposes is that it does not reflect joint limits. When the robot is near a joint limit, its movement is restricted. In an effort to include samples near joint limits we adopt the following convention: at configurations in which some joint is near a limit, the manipulability is defined to be zero. The nearness of a joint to its limit is a parameter of our sampling algorithm.

Example manipulability pdfs are shown in Figures 3-5. As can be seen in the figures, the manipulability pdfs tend to be unimodal, and quite smooth.

For our implementation, we define the manipulability measure to evaluate the ability of the robot to change the position of the end-effector (i.e., we use the manipulator Jacobian that relates joint velocities to the linear velocity of the end effector). In these figures, the plots labeled "Not filtered" correspond to sampling the manipulability of the robot without filtering out samples in which the robot is in self-collision; the plots labeled "Filtered" do exclude these samples. In all cases, 10 million samples were evaluated for manipulability. In addition, the gnuplot "csplines" function was used to smooth the plots. An explanation for the shift that can be seen for both robots in the probability distribution when filtering out self collisions is that configurations in which the robot is in collision with itself tend to be configurations for which the manipulability is low.

4 Results

To evaluate sampling biased by manipulability, we used a modified form of the planner for planar fixed-

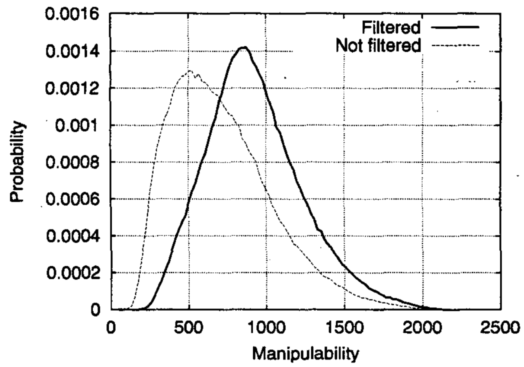

Figure 3: The pdf for the manipulability of a planar robot with six joints.

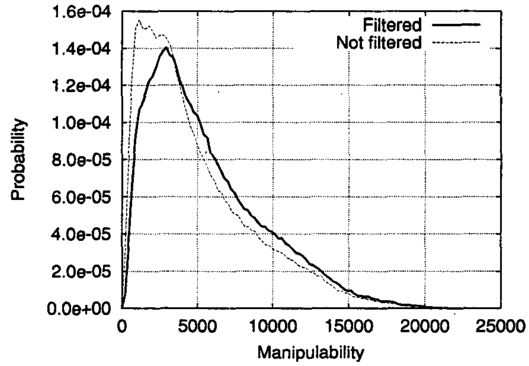

Figure 4: The pdf for the manipulability of a robot with six joints in a 3D workspace.

based articulated robots described in [9]. In particular, we added a function to the preprocessing phase to compute whether to reject a configuration based on its manipulability. We further modified the planner to adjust the order in which tests are applied to a random sample of the configuration space to determine whether to accept a sample. For each random sample, we test first whether the robot is in self-collision, then we apply the manipulability bias criterion, and last test the sample for collision between the robot and the obstacles. If the sample passes all tests, it is added to the network. The remainder of the preprocessing phase continues as described in [9].

To evaluate the planner, we performed a similar set of experiments to those described in [9]: for each set of parameters, we generate 40 networks and then test whether eight test configurations, shown in Figure 6, can be connected to the network. As a baseline, we include the results using unbiased sampling.

An explanation for the labels on the tables is as follows: The columns marked "Nodes" represents the target number of nodes for the roadmap after preprocessing, with "N" nodes generated during random sampling and "M" nodes generated during enhancement. The columns labeled "Number Rejected" list the number of nodes that failed a test: robot self-collision ("Self"), manipulability bias ("Manip"), and robot collision with an obstacle ("Obstacle"). The next three columns show three more statistics for the preprocessing phase. The column labeled "Avg. Size" lists the average size of the largest connected component in the roadmap after preprocessing. The column labeled "Avg. Comps" lists the average number of components in the roadmap after preprocessing, and the column labeled "Avg. Time" lists the average processing time required by the preprocessing phase. The last columns show the success rate over the 40 roadmaps of connecting the configurations shown in Figure 6.

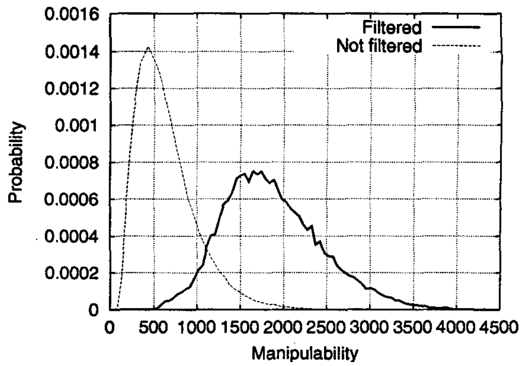

Figure 5: The pdf for the manipulability of a planar robot with twenty joints.

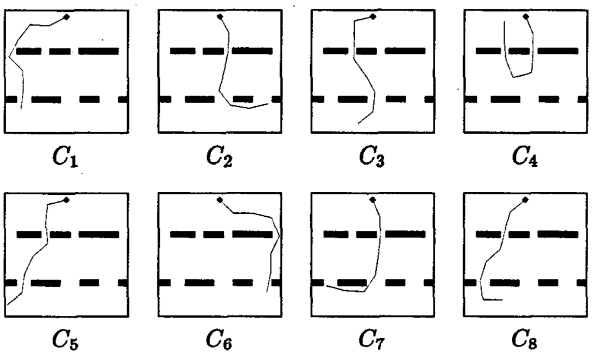

Figure 6: Eight configurations of a 7-revolute-joint fixed-base robot.

Table 1: Results for unbiased sampling with enhancement.

Nodes		Number Rejected			Avg.	Avg.	Avg	Connection success rate (%)							
N	M	Self	Manip	Obstacle	Size	Comps	Time	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
800	400	94367	0	43799	911	59	9.604	100	63	60	60	60	100	65	58
1000	500	116774	0	54253	1253	52	12.978	100	78	78	78	78	100	78	78
1200	600	140434	0	65137	1584	48	16.559	100	88	88	88	90	100	88	88
1400	700	163250	0	75763	1916	43	20.338	100	98	90	98	93	100	98	90
1600	800	185751	0	86218	2240	42	24.157	100	98	100	98	100	100	98	100
1800	900	209249	0	97039	2534	40	28.108	100	100	98	100	98	100	100	98
2000	1000	232937	0	108022	2862	37	32.126	100	100	100	100	100	100	100	100
2200	1100	256321	0	118841	3144	35	36.323	100	100	100	100	100	100	100	100
2400	1200	279882	0	129942	3456	34	40.626	100	100	100	100	100	100	100	100
2600	1300	302125	0	140305	3747	32	44.903	100	100	100	100	100	100	100	100
3000	1500	347880	0	161483	4359	31	53.779	100	100	100	100	100	100	100	100

Table 2: Results for sampling biased toward high manipulability with enhancement.

Nodes		Number Rejected			Avg.	Avg.	Avg	Connection success rate (%)							
N	M	Self	Manip	Obstacle	Size	Comps	Time	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
800	400	157374	82433	32451	881	63	10.543	100	38	35	35	35	100	40	35
1000	500	195695	102337	40384	1231	57	14.159	100	63	70	60	70	100	60	68
1200	600	236037	123329	48690	1549	52	17.915	100	75	80	75	90	100	75	80
1400	700	275244	143893	56812	1854	50	21.877	100	85	85	83	85	100	83	85
1600	800	315373	164900	65038	2175	45	25.822	100	88	93	88	93	100	88	95
1800	900	354719	185277	73182	2520	45	30.074	100	100	98	100	98	100	100	98
2000	1000	391648	204513	80872	2816	41	33.991	100	95	98	95	98	100	95	98
2200	1100	431421	225607	89066	3123	39	38.417	100	98	100	98	100	100	98	100
2400	1200	470227	245733	97018	3441	36	42.782	100	100	100	100	100	100	100	100
2600	1300	511244	267431	105446	3760	35	47.204	100	100	100	100	100	100	100	100
3000	1500	588644	307629	121385	4359	34	55.602	100	100	100	100	100	100	100	100

5 Discussion and Conclusions

We begin by noting that some of our results in Table 1 are slightly better than those originally reported in [9]. This can be attributed to improvements in computing power since those early results were published.

It can be seen in Tables 2, 3, 5 and 6 that our new approach is significantly more selective than unbiased approaches. Our manipulability-based rejection criterion rejects 2 to 3 times the number of nodes as are rejected due to collision with obstacles. Thus, one can see from these tables the trade-off between efficacy in node selection and the amount of computation required to construct the PRM.

By comparing Tables 1 and 6, it can be seen that using manipulability-biased sampling *without* enhancement produces PRMs that are nearly as effective as those that are produced by unbiased sampling *with* enhancement. This indicates that it may be possible to drive PRM enhancement using primarily intrinsic properties of the robot arm, as opposed to properties that are specific to the obstacles in a given workspace. This opens the door for new representations that can be constructed for arbitrary workspaces, as in some of our related work [15, 14].

Based on these results, we believe that our new approach to biased sampling can play a useful role in the construction of PRMs for many path planning applications.

References

[1] J. M. Ahuactzin, K. Gupta, and E. Mazer. Manipulation planning for redundant robots: A practical approach. *International Journal of Robotics Research*, 17(7):731–747, July 1998.

[2] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo. OBPRM: An obstacle-based PRM for 3D workspaces. In *Proceedings of Workshop on Algorithmic Foundations of Robotics*, pages 155–168, 1998.

[3] N. M. Amato and Y. Wu. A randomized roadmap method for path and manipulation planning. In *Proceedings of IEEE Conference on Robotics and Automation*, volume 1, pages 113–120, 1996.

[4] V. Boor, M. H. Overmars, and A. F. van der Stappen. The gaussian sampling strategy for probabilistic

Table 3: Results for sampling biased toward low manipulability with enhancement.

Nodes		Number Rejected			Avg.	Avg.	Avg	Connection success rate (%)							
N	M	Self	Manip	Obstacle	Size	Comps	Time	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
800	400	228219	96440	58891	970	43	10.793	98	75	85	75	85	100	75	85
1000	500	287113	121424	74035	1366	37	14.509	100	95	98	95	98	100	95	98
1200	600	341759	144396	88094	1651	36	18.473	100	95	98	95	98	100	95	98
1400	700	403436	170477	103999	1955	36	22.697	100	95	100	95	100	100	95	100
1600	800	460660	194606	118832	2275	33	26.885	100	100	100	100	100	100	100	100
1800	900	520620	219997	134298	2567	31	31.233	100	100	100	100	100	100	100	100
2000	1000	576715	243793	148752	2868	28	35.729	100	100	100	100	100	100	100	100
2200	1100	633438	267774	163294	3156	28	40.162	100	100	100	100	100	100	100	100
2400	1200	688248	290849	177390	3463	27	44.795	100	100	100	100	100	100	100	100
2600	1300	746051	315498	192478	3758	25	49.453	100	100	100	100	100	100	100	100
3000	1500	863813	365067	222769	4348	25	59.231	100	100	100	100	100	100	100	100

Table 4: Results for unbiased sampling without enhancement.

Nodes		Number Rejected			Avg.	Avg.	Avg	Connection success rate (%)							
N	M	Self	Manip	Obstacle	Size	Comps	Time	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
1200	0	139751	0	64889	680	168	8.762	100	8	5	8	8	100	8	3
1500	0	174232	0	80865	966	167	12.047	100	23	30	23	33	100	23	35
1800	0	210632	0	97676	1269	175	15.531	100	40	48	40	45	100	40	45
2100	0	244919	0	113780	1595	177	19.186	100	58	60	58	58	100	60	58
2400	0	279583	0	129809	1944	176	22.923	100	65	78	65	75	100	65	75
2700	0	314987	0	146154	2207	180	26.698	100	70	73	70	73	100	70	73
3000	0	348680	0	161858	2622	186	30.643	100	88	90	88	88	100	88	88
3300	0	384748	0	178657	2902	185	34.759	100	85	90	85	90	100	85	90
3600	0	419583	0	194684	3226	187	38.802	100	88	98	88	98	100	88	98
3900	0	455130	0	211280	3569	192	42.954	100	95	98	95	98	100	95	98
4500	0	527326	0	244786	4201	194	51.536	100	100	100	100	100	100	100	100

roadmap planners. In *Proceedings of IEEE Conference on Robotics and Automation*, pages 1018–1023, 1999.

[5] L. J. Guibas, C. Holleman, and L. E. Kavraki. A probabilistic roadmap planner for flexible objects with a workspace medial-axis-based sampling approach. In *Proceedings of IEEE/RSJ Conference on Intelligent Robots and Systems*, 1999.

[6] C. Holleman and L. E. Kavraki. A framework for using the workspace medial axis in PRM planners. In *Proceedings of IEEE Conference on Robotics and Automation*, pages 1408–1413, 2000.

[7] T. Horsch, F. Schwarz, and H. Tolle. Motion planning with many degrees of freedom — random reflections at c-space obstacles. In *Proceedings of IEEE Conference on Robotics and Automation*, pages 3318–3323, 1994.

[8] D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. *International Journal of Computational Geometry and Applications*, 9(4 & 5):495–512, 1999.

[9] L. E. Kavraki. *Random Networks in Configuration Space for Fast Path Planning.* PhD thesis, Stanford University, Stanford, CA, 1994.

[10] L. E. Kavraki, M. N. Kolountzakis, and J.-C. Latombe. Analysis of probabilistic roadmaps for path planning. In *Proceedings of IEEE Conference on Robotics and Automation*, volume 4, pages 3020–3025, 1996.

[11] L. E. Kavraki and J.-C. Latombe. Randomized preprocessing of configuration space for fast path planning. In *Proceedings of IEEE Conference on Robotics and Automation*, volume 3, pages 2138–2145, 1994.

[12] L. E. Kavraki and J.-C. Latombe. Probabilistic roadmaps for robot path planning. In K. Gupta and P. del Pobil, editors, *Practical Motion Planning in Robotics: Current Approaches and Future Directions*, pages 33–53. John Wiley & Sons LTD, 1998.

[13] J. J. Kuffner, Jr. and S. M. LaValle. RRT-connect: An efficient approach to single-query path planning. In *Proceedings of IEEE Conference on Robotics and Automation*, pages 995–1001, 2000.

[14] P. Leven and S. Hutchinson. Real-time path planning in changing environments: Some preliminary results. In *International Symposium on Robotics*, 2000.

[15] P. Leven and S. Hutchinson. Toward real-time path planning in changing environments. In *Proceedings*

Table 5: Results for sampling biased toward higher manipulability without enhancement.

Nodes		Number Rejected			Avg.	Avg.	Avg	Connection success rate (%)							
N	M	Self	Manip	Obstacle	Size	Comps	Time	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
1200	0	235671	123258	48577	809	178	10.104	100	3	5	0	0	100	0	5
1500	0	293136	153299	60512	1055	182	13.590	100	3	15	3	13	100	3	13
1800	0	353514	184734	72905	1285	186	17.139	100	8	10	5	13	100	5	10
2100	0	412814	215555	85175	1556	189	20.905	100	23	15	20	18	100	20	18
2400	0	468641	244993	96665	1855	197	24.682	100	40	28	40	35	100	40	30
2700	0	531315	277513	109491	2109	198	28.737	100	35	38	35	38	100	35	38
3000	0	590763	308767	121869	2434	199	32.658	100	53	48	53	48	100	55	48
3300	0	646986	338165	133384	2691	207	36.673	100	50	50	50	53	100	50	53
3600	0	705373	368756	145486	3075	204	40.932	100	65	73	65	78	100	65	73
3900	0	767938	401333	158583	3329	206	45.224	100	73	60	73	58	98	73	60
4500	0	882189	461151	181934	4011	212	54.159	100	88	70	88	70	100	88	70

Table 6: Results for sampling bias toward lower manipulability without enhancement.

Nodes		Number Rejected			Avg.	Avg.	Avg	Connection success rate (%)							
N	M	Self	Manip	Obstacle	Size	Comps	Time	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
1200	0	342290	144614	88228	605	136	10.558	80	48	50	48	50	60	48	50
1500	0	429853	181626	110792	913	135	14.482	93	48	63	48	63	80	48	63
1800	0	515776	217943	132974	1367	142	18.474	98	65	83	65	83	98	65	83
2100	0	599852	253770	154705	1647	148	22.638	100	68	85	68	85	98	68	85
2400	0	688392	290852	177466	2085	150	26.852	100	85	95	85	95	100	85	95
2700	0	771972	326285	199132	2472	152	31.224	100	98	98	98	98	100	98	98
3000	0	863181	364893	222513	2709	156	35.763	100	90	100	90	100	100	90	100
3300	0	945738	399917	243895	3092	157	40.265	100	100	100	100	100	100	100	100
3600	0	1031846	436321	266026	3385	159	44.754	100	100	100	100	100	100	100	100
3900	0	1119291	473301	288500	3678	163	49.450	100	100	100	100	100	100	100	100
4500	0	1291941	546276	333038	4258	170	59.148	100	100	100	100	100	100	100	100

of Workshop on Algorithmic Foundations of Robotics, 2000.

[16] M. H. Overmars and P. Švestka. A probabilistic learning approach to motion planning. In *Proceedings of Workshop on Algorithmic Foundations of Robotics*, pages 19–37, 1994.

[17] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. Motion planning for a rigid body using random networks on the medial axis of the free space. In *Proceedings of ACM Symposium on Computational Geometry*, pages 173–180, 1999.

[18] T. Yoshikawa. Manipulability of robotic mechanisms. *International Journal of Robotics Research*, 4(2):3–9, Apr. 1985.

[19] Y. Yu and K. Gupta. Sensor-based roadmaps for motion planning for articulated robots in unknown environments: Some experiments with an eye-in-hand system. In *Proceedings of IEEE/RSJ Conference on Intelligent Robots and Systems*, 1999.