Adaptive Strategies for Probabilistic Roadmap Construction

P. Isto

Industrial Information Technology Laboratory
Helsinki University of Technology
P.O. Box 9204, 02015 HUT
Pekka.Isto@hut.fi

J. Tuominen

Industrial Information Technology Laboratory
Helsinki University of Technology
P.O. Box 9204, 02015 HUT
Juha.Tuominen@hut.fi

M. Mäntylä

Helsinki Institute for Information Technology
P.O. Box 9800, 02015 HUT
Martti.Mantyla@hiit.fi

Abstract

This paper presents an experimental study of prospects for using adaptable local search techniques in probabilistic roadmap based motion planning. The classical PRM approach uses a single fast and simple local planner to build a network representation of the configuration space. Advanced PRM planners utilize heuristic sampling techniques and combine multiple local planners. The planner described here uses a single local planner, but adjusts its competence during the roadmap construction stage according to the problem at hand. Two adjusting strategies are proposed and compared experimentally against using a static local planner at a set competence level. The results indicate that roadmap construction with an adaptive local planner can bring advantages including more robust performance and a reduction in planning cost variance.

1. Introduction

Motion planning for robots and other devices is an interesting, but computationally hard problem [1]. Many motion planning algorithms have been presented in the literature, but due to the complexity of the problem, no single generally applicable motion planner exists. Practical motion planning becomes very difficult and heuristic and incomplete algorithms must be used if the problem has more than a few degrees-of-freedom [2]. Most contemporary motion planning algorithms are randomized and trade the completeness of the algorithm for improvement in expected performance.

Probabilistic roadmap planners are a family of randomized motion planning algorithms that have been a subject of intensive theoretical and experimental research. The roadmap approach is a global approach that aims to build a representation of the connectivity of the configuration space (*cspace*) as a network of curves [3]. Probabilistic roadmap (PRM) planners use randomized sampling of the *cspace* and local planning between the samples to produce a probabilistic approximation of the connectivity of the *cspace*. PRMs can be used in two modes. In the first mode, a separate learning or preprocessing stage builds the roadmap and motion planning queries for the preprocessed problem are answered with the help of the roadmap. In the second mode, the roadmap is built during the query until it contains a solution for the query. In the latter mode, the roadmap is usually discarded immediately after the query is answered. A particular algorithm is often tuned for one or the other mode, but many algorithmic techniques can be used in both types of PRM planning.

Since the motion planning problem is PSPACE-hard [3], it is difficult to design practical planners that have a consistently good performance over a range of different tasks. Many techniques have been presented to improve the performance of motion planners, but many tend to rely on assumptions on the structure of the task, which are easy to violate. Indeed, the complexity of the problem guarantees the existence of worst-case counter-examples for any heuristics. Combining multiple heuristics to be used simultaneously and using run-time selection between those heuristics can yield more robust planners, which retain most of the efficiency of the individual heuristics.

This paper investigates the possible benefits of using adaptive local search techniques in PRM planning. The approach has motivations similar to those for combining multiple local planners, but the planner described here uses only a single local planner. An effect similar to combining multiple local planners is obtained by adjusting the competence of the single local planner during the roadmap construction stage according to the problem at hand. As the capability of the local planner can vary widely and it can be controlled in a continuous way, there is no need to combine several planners with different properties and select among them. The same local planner can be a fast but incomplete local planner for easy tasks or a slow but theoretically resolution-complete planner for difficult tasks.

The dependence of the planning cost on the capability of the local planner is studied by producing an empirical cost surface for a well-known test problem. Since the cost surface appears to have exploitable structure, the obvious question is how to control the capability of the local planner in order to exploit the structure for best possible performance. Two simple formulas for adjusting the local planner are presented and evaluated in this paper.

The following sections describe relevant previous research, introduce the PRM planner and test problems used in this paper, and present the results. The final section presents the conclusions. The research presented in this paper is empirical using experiments and statistical analysis of the results.

2. Previous Work

The probabilistic roadmap planning was introduced simultaneously by several research groups [4][5][6]. A central idea of the approach is to use a graph structure to represent the connectivity of the *cspace* and solve motion planning problems by connecting the start and goal configurations to the graph. The graph structure – or roadmap – is constructed by sampling collision-free configurations and producing path segments between the samples with a fast and simple local planner. Various criteria such as distance or visibility [7] can be used to select the sample pairs submitted to the local planner.

The simplest PRM planners rely heavily on randomization to provide opportunities for the local planner to capture the connectivity of the *cspace*. The simple approach is susceptible to the "narrow passage" problem [1]. If the solution requires passing a bottleneck, the probability of finding samples allowing the local planner to find a path through the bottleneck can be made arbitrarily small by narrowing the bottleneck. Another problem with the PRM approach is the variation of solution cost and quality from run to run. Various methods have been presented for addressing the narrow passage problem, e.g. [8],[10],[11], but the run-time variance is largely an unaddressed problem.

The difficulty of developing consistently fast and robust motion planners has motivated some researchers to consider combining several different planners together and selecting one for execution depending on the problem at hand. Hwang proposes combining a fast but incomplete heuristic planner with a slower but complete planner [12]. The fast planner is run first and, if it fails to solve the problem, the complete planner is invoked to solve the problem. Amato *et al.* propose to combine several simple local planners in sequential stages during the roadmap construction [9]. Vallejo *et al.* describe a framework for the use of several different local planners and subgoal generation methods for single query planning [13]. A technique for controlling the competence of the local planner was introduced in [14], but presented in conjunction with a complex and itself dynamic local planner and not studied in any detail. This paper can be seen as a further study of those ideas. Generally, criteria for selecting motion planners to combine and techniques for selecting which one to execute at a given time are largely open research problems, although some ideas have been suggested [13].

Most of the above research is empirical in nature. However, a number of theoretical results form the theoretical foundation of PRM planners. In brief, the probability of failing to find a path from a probabilistic roadmap decreases exponentially as the number of samples in the roadmap is increased [15]. Additional results can be found in the literature [16].

3. An Adjustable Roadmap Planner

The experimental framework of this paper includes a simple PRM planner that uses a bi-directional A^* search [17] based local planner. Although A^* search algorithm is complete, it is not suitable for motion planning alone due to exponential memory consumption. However, a measure for controlling the search efficiency of A^* can be easily obtained from the ratio of the size of the examined search space to the length of the best available path candidate towards the target sample. This measure is very similar to Nilsson's penetrance [17], and on grid representation of the *cspace*, it can be realized with simple counting operations. Let $F(C)$ be the total number of collision-free configurations examined by the local planner until the examination of configuration C. Furthermore, let $g(C)$ be the distance from the start sample to the configuration C currently examined by the local planner. The efficiency measure can be defined as

$$O(C) = \frac{F(C)}{g(C)}$$

The competence of the local planner can be controlled by setting an upper limit O_{th} for $O(C)$ and discontinuing the search in the local planner, if the limit is exceeded. Thus, the local planner fails if the search becomes inefficient. This also controls the memory consumption by setting an indirect limit on the amount of memory used during the search.

The guiding function for A^* is $f(C) = A \times g(C) + B \times h(C)$, where $g(C)$ is as above, $h(C)$ is a heuristic estimate for the cost from the currently examined configuration C to the target sample. The guiding function used here is greedy with constants A=3 and B=5. Manhattan distance in the discrete *cspace* is used for both $g(C)$ and $h(C)$. The heuristic estimate $h(C)$ has an additional tie breaking term that favors motions, which move the same degree-of-freedom repeatedly.

The sampling strategy used here is simple pseudo-random sampling until a roadmap connecting all the seed configurations of the given test task is obtained. Candidate sample pairs for the local planner are produced by selecting for the newly generated sample up to $k=10$

closest nodes from each connected component of the roadmap at the sample generation time. Euclidean distance is used as the distance metric in the selection.

The central part of this paper is the question of how to determine the threshold for *O(C)*. The majority of PRM planners have static local planners, which can be obtained for this study by setting a fixed upper limit for *O(C)*. Such planners are named here with a "*A*" prefix followed by the numerical value for the *O(C)* limit.

It may be possible to use information gained during the roadmap construction to determine a particular value of *O(C)* for each call of the local planner. Two such adaptation strategies are studied experimentally in this paper. The first strategy involves increasing the *O(C)* limit linearly with the size of the roadmap. The intuition behind this strategy is that more difficult problems require larger roadmap and a more capable local planner to adequately describe the connectivity of the *cspace*. Furthermore, as more samples are added to the roadmap, the failure probability of the local planner decreases [15]. With lower failure probability, failures that are more expensive can be tolerated. This strategy has a global character in the sense that it determines a single increasing *O(C)* limit for all the samples in the roadmap. The following formula is used to determine the threshold O_{th} for *O(C)* during roadmap construction at a particular roadmap size of *S*:

$$O_{th}(s) = \frac{S}{s} \times 32 \text{ .}$$

A number of such strategies are defined with various values for the constant *s*. The constant of 32 is the largest static limit used in the experiments. The strategies are named with a "*AL*" prefix followed by the numerical value for *s*.

A local strategy is defined by setting the O_{th} threshold separately for each sample in the roadmap. The strategy uses a measure of difficultness of the *cspace* around a particular sample. For each sample *v* the fraction of successful calls of local planner is computed:

$$r_s(v) = \frac{N_s(v)+1}{N(v)+1} \text{ ,}$$

where *N(v)* is the total number of local planner calls with the configuration space sample *v* either as start or target and $N_s(v)$ is the number of calls that succeeded in producing a path segment to or from the sample *v*. This measure is very similar to failure ratio [18].

A value of O_{th} is computed for both start and target samples with the parameterized formula:

$$O_{th}(v,n) = 1 + \frac{n}{r_s(v)} \text{ ,}$$

and the maximum is used as the current threshold value. The constant *n* is used to define a family of strategies, each named with a "*AN*" prefix followed by the numerical value for *n*. The intuition behind this strategy is that a large failure ratio suggests that node resides in a difficult region of the *cspace* and thus the local planner should be given opportunity to search the region more broadly.

The PRM planners described above are quite powerful ones capable of capturing the connectivity of the Alpha Puzzle 1.2 benchmark problem [9] in tens of minutes and even solving the very difficult Alpha Puzzle 1.0 with a 500 MHz PC. However, a deployable motion planner would certainly use a more efficient guiding heuristics and more sophisticated sampling strategy such as those described in section 2. The aim here is not to present a new or state-of-the-art motion planning algorithm, but to study the behavior of adjusting strategies in simple and straightforward setting.

4. The Test Problems

The test problems are two well-know benchmark problems proposed in the literature. The Hwang and Ahuja benchmark problem is a 5 degrees-of-freedom robotics motion planning problem for a SCARA-type robot [3]. The task was designed to represent a realistic but non-pathological problem for a manipulator. The task involves removing a hook from a wicket and a subsequent backtracking motion to avoid a large obstacle (see Fig. 1). No generally available geometric model for the task exists, but several versions of the problem were made for this study. The versions differ in the depth of the notch made in the L-shaped large obstacle. The depth controls the amount of free-space available in between of the high obstacle and the robot body. The narrower the free-space the more difficult the problem is, since the motion planner must find a path through the bottleneck. The problem versions are named with *Adept* prefix followed by a numerical value for the depth of the notch in millimeters. The problem becomes unsolvable around the depth of 75mm.

The second test problem is the Alpha Puzzle benchmark problem proposed by Amato *et al.* [9]. The problem is intended to represent 6 degrees-of-freedom disassembly problems and it is designed to have a narrow passage. Several versions of the Alpha Puzzle exist with varying difficultiness. A smaller version number indicates more difficult problem. The original Alpha Puzzle problem involves separating the two intertwined loops. The loops can be intertwined in two different ways with the prongs of the loops either in symmetric (first image in figure 2) or anti-symmetric (last image in figure 2) orientations. Since the intention of this paper is to evaluate the performance of the various strategies in capturing the full connectivity of the test problems, both intertwined configurations together with a separated configuration (middle image in figure 2) are inserted as seed configurations to the roadmap at the beginning of the construction. Thus, the final roadmaps must connect all the three seed configurations in the figure 2.

5. Empirical Results

First, the effect of the threshold O_{th} is investigated experimentally to determine if any systematic response can be observed. Roadmaps are constructed for the test problems with the PRM planner until a roadmap connecting the seed configurations is obtained. The number of collision checks performed during the roadmap construction is used as a cost measure. Tables 1 and 2 present the average construction costs for the test problems at various fixed levels for the threshold. Additionally, table 1 presents construction costs for a naive PRM planner, which tries to connect the nodes with a simple straight-line local planner (labeled *SL*). The data from table 1 for the versions of Hwang and Ahuja problem is visualized as a surface in figure 3. It can be seen that the performance of the PRM planner depends strongly on the threshold value, especially for the more difficult versions of the problem. Furthermore, there is a minimum cost "valley" on the surface across the versions of the problem. The interpretation of the data suggests that the best value for the threshold depends on the difficulty of the problem with higher values preferred for more difficult problems. The data in table 2 presents average roadmap construction costs for the Alpha Puzzle benchmarks. The Alpha Puzzle data indicates behavior similar to the one observed for the Hwang and Ahuja problem, but here the valley seems to locate at higher levels of O_{th}. No solutions could be produced with the *A1* strategy for Alpha Puzzle versions 1.0 and 1.1, and with *A2* strategy for Alpha Puzzle 1.0. Additional supporting observations of the association between the capability of the local planner and the solution cost were made in previous experiments with different problems and heuristics [14].

The finding of a possible minimum cost valley in the cost surface motivates development of techniques which can take advantage of the information gathered during the roadmap construction and utilize that information to reduce the construction cost. The adaptive strategies described above are steps to this direction as they use measures of the difficultiness of the problem to adjust heuristically the local planner. Due to computational cost, the comparison of the adaptive strategies against static ones is restricted to *Adept80* and Alpha Puzzle 1.2 problems.

Table 3 displays the average numbers of collision checks and their coefficients of variation for each of the strategies over a range of relevant parameters. The static O_{th} threshold is varied over a range from 2 to 32, the parameter s for the linear strategy from 3×10^2 to 90×10^2 and the parameter n for the node strategy from 1×10^{-2} to 30×10^{-2}. The static *A1* strategy is excluded from this experiment due to the excessive cost observed with Alpha Puzzle 1.2 (see table 2). Investigation of the data for *Adept80* reveals that while the static strategy is very sensitive for the value of the O_{th} threshold, the adaptive strategies are remarkably robust for the selection of the parameters s and n. When accounting for the scale difference, the range of s and n values is almost twice the range of the static O_{th} threshold. Furthermore, the coefficient of variation is smaller for the adaptive strategies. For the Alpha Puzzle 1.2, the behavior of the three strategies appears to be quite similar. This may be explained by the fact that it is very difficult to generate good *cspace* samples for this problem. The final roadmap has to have paths through two long narrow passages and the probability of getting samples in or near the passages is quite low. It would be interesting to test this hypothesis by repeating the experiment with a more advanced sampling method.

Based on the data and analysis presented here, the *AN0.1* strategy can be recommended for tasks similar to the test problems used here. *AN0.1* has smallest average cost with relatively small coefficient of variation for this version of the Hwang and Ahuja problem and its performance is also among the best for Alpha Puzzle 1.2. If a problem with unknown properties must be solved, one of the adaptive strategies should be used since they are more robust for the exact values of the heuristic parameters.

6. Conclusions

This paper presented an experimental study of using adaptable local search techniques in probabilistic roadmap based motion planning. An empirical cost surface of a well-known benchmark problem from the literature suggests that there are prospects for using information collected during the roadmap construction to reduce the over-all construction cost. Simple adaptation strategies were introduced and tested experimentally. Adaptation of the local planner during the construction can make a PRM based motion planning algorithm more robust and reduce the variance in the running cost. Therefore, adaptation strategies should be used in combination with search based local planners. More research is needed to investigate if more advanced sampling techniques can provide synergetic improvement in the efficiency of PRM motion planners.

Acknowledgments

Alpha Puzzle was designed by Boris Yamrom, GE Corporate Research & Development Center. The model was provided by the DSMFT research group at Texas A&M University. Johannes Lehtinen provided the

geometric model for the Hwang and Ahuja task. Janne Ravantti provided access to the cluster computer at the Bamford Laboratory, University of Helsinki.

References

[1] J.-C. Latombe, Motion Planning: A Journey of Robots, Molecules, Digital Actors, and Other Artifacts, International Journal of Robotics Research, Vol. 18, No. 11, Nov. 1999, 1119-1128.

[2] K. Gupta, Overview and State of the Art, in K Gupta, A. P. del Pobil (eds.), Practical Motion Planning in Robotics: Current Approaches and Future Directions, John Wiley & Sons, West Sussex, 1998.

[3] Y. K. Hwang and N. Ahuja, Gross Motion Planning - A Survey, ACM Computing Surveys, Vol. 24, No. 3, Sep. 1992, 219-291.

[4] Th. Horsch, F. Schwarz and H. Tolle, Motion Planning with Many Degrees of Freedom – Random Reflections at C-Space Obstacles, Proceedings of the 1994 IEEE International Conference on Robotics and Automation, IEEE, 3318-3323.

[5] L. Kavraki and J.-C. Latombe, Randomized Preprocessing of Configuration Space for Fast Path Planning, Proceedings of the 1994 IEEE International Conference on Robotics and Automation, IEEE, 2138-2145.

[6] M. H. Overmars and P. Svestka, A Probabilistic Learning Approach to Motion Planning, Technical Report UU-CS-1994-03, Department of Computer Science, Utrecht University, The Netherlands, January 1994.

[7] C. Nissoux, T. Simeon, and J.P. Laumond, Visibility Based Probabilistic Roadmaps, Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 1316-1321.

[8] V. Boor, M. H. Overmars, and F. van der Stappen, The Gaussian Sampling Strategy for for Probabilistic Roadmap Planners, Proceedings of the 1999 IEEE International Conference on Robotics and Automation, IEEE, 1018-1023.

[9] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, OBPRM: An Obstacle-Based PRM for 3D Workspaces, Workshop on Algorithmic Foundations of Robotics, 1998, 155-168.

[10] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, MAPRM: A Probabilistic Roadmap Planner with Sampling on the Medial Axis of the Free Space, Proceedings of the 1999 International Conference on Robotics and Automation, IEEE, 1024-1031.

[11] P. Isto, Constructing Probabilistic Roadmaps with Powerful Local Planning and Path Optimization. Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2323-2328.

[12] Y. K. Hwang, Completeness vs. Efficiency in Real Applications of Motion Planning, Proceedings of the Workshop on Practical Motion Planning in Robotics: Current Approaches and Future Directions, Minneapolis, MN, IEEE, 1996.

[13] D. Vallejo, C. Jones, and N. M. Amato, An Adaptive Framework for `Single Shot' Motion Planning, Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 1722-1727.

[14] P. Isto, A Two-level Search Algorithm for Motion Planning, In: Proceedings of the 1997 IEEE International Conference on Robotics and Automation, IEEE, 2025-2031.

[15] L. Kavraki, J.-C. Latombe, R. Motwani, and P. Raghavan, Randomized Query Processing in Robot Motion Planning, Proceeding of the ACM SIGACT Symposium on Theory of Computing (STOC), ACM, Las Vegas, 1995, 353-362.

[16] D. Hsu, L. E. Kavraki, J.-C. Latombe, and R. Motwani, Capturing the Connectivity of High-Dimensional Geometric Spaces by Parallelizable Random Sampling Techniques, in P. M. Pardalos and S. Rajasekaran (eds), Advances in Randomized Parallel Computing, Combinatorial Optimization Series, Kluwer Academic Publishers, Boston, 1999, 159-182.

[17] N. J. Nilsson, Principles of Artificial Intelligence, Springer-Verlag, Berlin, 1982.

[18] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. IEEE Transactions on Robotics and Automation, Vol. 12, No. 4, 1996, 566-580.

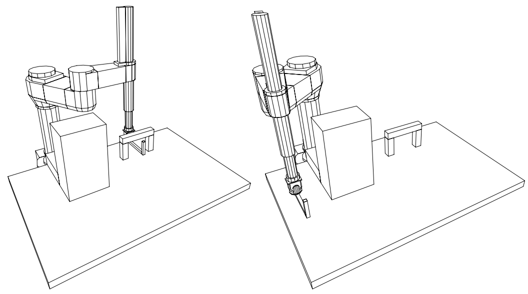

Fig. 1. The seed configurations for a version of the benchmark problem proposed by Hwang and Ahuja. This figure presents the easiest version with passage depth of 400 mm.

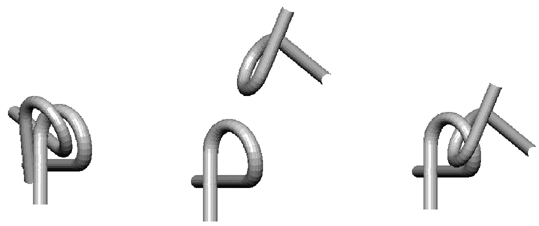

Fig. 2. The seed configurations for the Alpha Puzzle benchmark task.

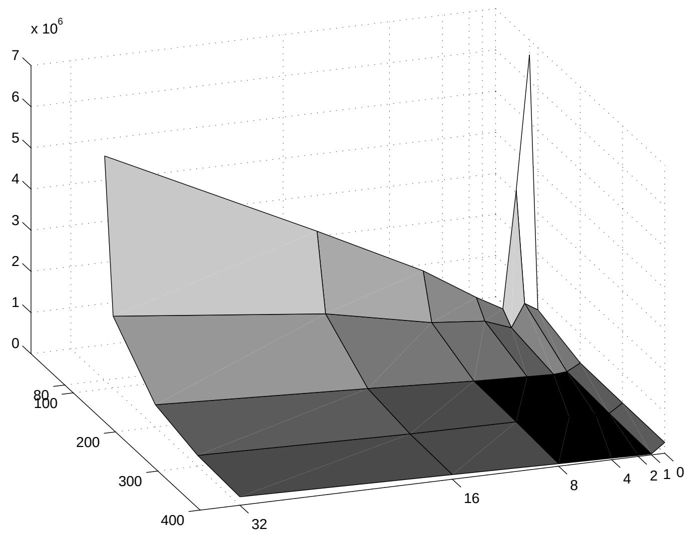

Fig. 3. Cost surface for various versions of the Hwang and Ahuja benchmark problem.

	A32	A16	A8	A4	A2	A1	SL
Adept80	5,446,407	2,986,472	1,706,875	904,616	548,688	3,389,542	6,642,256
Adept100	1742,697	1170,693	640,290	517,955	279,917	830,774	635,328
Adept200	542,358	301,870	167,786	120,968	95,044	123,068	292,904
Adept300	261,297	150,627	129,283	86,150	50,768	58,584	272,631
Adept400	203,713	112,787	61,117	38,961	25,500	23,974	259,681

Table 1. The average numbers of collision checks performed by the PRM planners for the versions of Hwang and Ahuja benchmark problem. Each data point represents an average of at least 15 runs.

	A32	A16	A8	A4	A2	A1
AP 1.0	226,692,577	138,243,470	94,844,153	76,840,334	—	—
AP 1.1	69,032,080	41,777,409	32,771,501	65,773,256	72,028,029	—
AP 1.2	1,729,588	1,224,763	1,127,312	1,166,626	1,908,043	156,882,356
AP 1.5	343,581	220,483	144,205	110,641	85,539	192,598

Table 2. Average construction costs (collision checks) for the versions of Alpha Puzzle problem. The sample size is at least 15 runs.

	Adept80		Alpha Puzzle 1.2	
	Cost	C. of Var.	Cost	C. of Var.
A2	638,781	90.7	1,908,043	68.5
A4	936,468	101.2	1,166,626	78.1
A8	1,530,012	107.3	1,127,312	60.2
A16	2,657,819	117.6	1,224,763	67.4
A32	4,343,757	131.2	1,729,588	70.8
AN0.01	1,224,995	73.7	1,440,078	50.2
AN0.03	669,978	89.9	1,301,360	65.4
AN0.1	581,961	91.6	1,250,518	65.6
AN0.3	790,115	118.9	1,437,473	84.8
AL300	1,221,051	194.7	1,240,348	65.8
AL1000	754,649	141.1	1,284,902	67.2
AL3000	624,564	91.4	1,448,377	52.3
AL9000	619,194	90.7	1,919,915	68.5

Table 3. Average construction costs (collision checks) and coefficient of variation for the tested strategies. The sample size is 60 runs.