Adaptive Neighbor Connection using Node Characterization

Chinwe Ekenna, Shawna Thomas, Nancy M. Amato[⋆⋆⋆]

No Institute Given

Abstract. Sampling-based motion planning has been successful in planning the motion for a wide variety of robot types. An important primitive of these methods involves connecting nodes by selecting candidate neighbors and checking the path between them. Recently, an approach called Adaptive Neighbor Connection (ANC) was proposed to automate neighbor selection using machine learning techniques. It adaptively selects a neighbor connection strategy from a candidate set of options and learns which strategy to use by examining their success rates and costs.
In this work, we extend ANC's reward function by characterizing the types of nodes added (including information about their connectivity) after each connection attempt. In doing so, we gain insight into the nature of these nodes and their ability to improve roadmap quality. We also refine ANC's cost function by considering the computation time spent during each connection attempt so as to potentially learn faster and more efficient connectors. We show improved performance over selecting a single connection strategy and over ignoring node characterization during learning. We present results in a variety of 2D and 3D environments and are able to solve queries in half the time on average compared to the best single connection strategy and to the original ANC work.

1 Introduction

The motion planning problem involves finding a valid path (e.g., collision-free) for a movable object (e.g., robot, vehicle or protein) from a start configuration to a goal configuration in a given environment. Motion planning is an important component of many applications including robotics [29], medicine [1],

[⋆] This research supported in part by NSF awards CNS-0551685, CCF-0833199, CCF-0830753, IIS-0917266, IIS-0916053, EFRI-1240483, RI-1217991, by NSF/DNDO award 2008-DN-077-ARI018-02, by NIH NCI R25 CA090301-11, by DOE awards DE-FC52-08NA28616, DE-AC02-06CH11357, B575363, B575366, by THECB NHARP award 000512-0097-2009, by Samsung, Chevron, IBM, Intel, Oracle/Sun, by Award KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST), and by Schlumberger Faculty for the Future Fellowship.

[⋆⋆] C. Ekenna, S. Thomas, and N. M. Amato are with the Department of Computer Science and Engineering, Texas A&M University, College Station, TX, 77843, USA {cekenna,sthomas, amato}@cse.tamu.edu.

gaming/virtual reality [12, 32], protein folding [28], and search and rescue operations [8, 17]. Exact methods for motion planning become intractable as the complexity of the robot increases [7].

Sampling-based motion planners such as Probabilistic Roadmap Methods (PRMs) [16] and Rapidly-exploring Random Trees (RRTs) [18] were developed as efficient alternatives to other approaches. These methods construct an approximate model of the robot's valid motion space by building a graph containing representative feasible trajectories. One component of these methods is constructing edges which connect neighboring nodes in the graph that encode feasible trajectories between them. It is infeasible to simply attempt all $O(n^2)$ possible connections. Thus, there have been many proposed methods for locating candidate neighbors [9,22,26]. This raises the question, how do you decide which one to apply? What environments would they be suitable for and for what type of robots?

In previous work [9], we employed a machine learning technique to learn the appropriate connection strategy to employ for an environment which we termed Adaptive Neighbor Connection (ANC). ANC sets selection probabilities for the available connection methods and increases or decreases them over time based on past performance. A similar idea has been used successfully during the node generation phase of PRM [13].

Although, ANC improves connectivity and quality of roadmaps produced, ANC ignores the characteristics of nodes added to the roadmap but ANC$_{nc}$, uses this information to improve path planning for a variety of robots.

In this paper, we show that planning for robots can be further improved on by including additional factors about the connected nodes. For example, how do the nodes being added to the roadmap improve (or not) roadmap connectivity? By characterizing the types of nodes added and including information about their connectivity after each connection attempt, we gain insight about the best use of each connector and can better identify when a connector ceases being useful. These are important aspects of the space that we train this new ANC (which we term, ANC$_{nc}$) to use as a criteria to determine if the connector's performance is improving the global scope of the roadmap or not. Also, ANC did not consider actual cost used by each connector during connection attempts, and based on our analysis and experiments, we found this to be beneficial information to include in the learning process.

As shown in our results, ANC$_{nc}$ makes a significant improvement in the time needed to solve the query for our 2D and 3D environments, solving problems on average in half the time. ANC$_{nc}$ rapidly learns the best strategy to employ based on a trade-off between the node type being added and cost. It is able to adapt to changing sampling density as roadmaps are incrementally constructed. We compare ANC$_{nc}$ to other popular connection strategies and ANC in variety of environments. Our results show that adding key nodes early to the roadmap significantly affects the connectivity and its query solving ability. In all the problems studied, ANC$_{nc}$ shows significant reduction in the time required to solve representative queries (by half on average over both ANC and the best individual connection strategy.

2 Preliminaries and Related Work

In this section we describe the motion planning problem, properties of the planning space, the sampling-based motion planning approach, and existing work on adaptive strategies for such methods.

2.1 Motion Planning Preliminaries

A robot is a movable object whose position and orientation can be described by n parameters, or degrees of freedom (DOFs), each corresponding to an object component (e.g., object positions, object orientations, link angles, link displacements). Hence, a robot's placement, or configuration, can be uniquely described by a point $(x_1, x_2, ..., x_n)$ in an n dimensional space (x_i being the ith DOF). This space, consisting of all possible robot configurations (feasible or not) is called configuration space (C-Space) [21]. The subset of all feasible configurations is the free space (C-Free), while the union of the infeasible configurations is the blocked or obstacle space (C-Obst). Thus, the motion planning problem becomes that of finding a continuous trajectory in C-Free connecting start and goal pair of configurations.

2.2 C-Space Properties

Two configurations q and q' are visible to each other if a path or a continuous sequence of adjacent configurations exists between them. Two configurations q and q' are therefore connectable if there exists continuous sequence of configurations that are also visible to each other. In such a case, we define connectable (q, q') = true. We define a connected component CC of C-Free as a maximal subset of configurations $CC \subseteq C - Free$ such that $\forall q, q' \in CC$, $connectable(q, q') = true$. The C-Free configuration space may comprise of one or more connected components.

2.3 Probabilistic RoadMaps for Motion Planning

Probabilistic RoadMaps (PRMs) [16], a sampling-based motion planning approach, uses a two stage process: roadmap construction and query processing. During roadmap construction, PRMs sample the configuration space, retaining valid ones, and attempt to connect them using some local planner. The probability of failing to find a path when one exists decreases exponentially as the number of samples in the roadmap increases [15]. There have been a number of methods proposed for locating candidate neighbors for connection because it is intractable to simply attempt all possible connections (i.e., $O(n^2)$ attempts). In [11], the maximal connectivity achieved by different sampling methods was compared to that of C-space being modeled. Maximal connectivity ensures that a path between them can be found in the roadmap. The authors evaluate the time needed to adequately cover and connect the C-free for various techniques. They describe the properties of these neighbor finding approaches and motivate research on connections based upon reachability analysis. This work relies on a discretization of C-space and so it cannot be applied to high DOF problems.

Since there is no principled mechanism to determine when to stop roadmap construction, a commonly used evaluation criterion is to predefine a set of relevant queries in each environment and continue building the roadmap until the query configurations can be connected to the same connected component. This is helpful in environments where the user knows beforehand such a representative query. Below is a description of some intelligent and adaptive strategies that have been employed to improve the solutions motion planning algorithms already proposed.

2.4 Adaptive Motion Planning Frameworks

Feature-Sensitive motion planning [24] uses machine learning to help partition and characterize a planning problems. In this approach, the planning space is subdivided in a recursive manner. Then each region is classified and assigned an appropriate planning method. One main strength of this approach is its ability to identify certain topology of the workspace or C-space for a particular planner to generate nodes in. It is not able to adapt planners overtime. This method also does not take care of how the nodes are connected in it's subdivided regions or if the nodes generated in these regions are good candidate neighbors for each other.

Hybrid PRM [13] uses a reinforcement-learning approach to provide sampler adaption by selecting a node generation method that is expected to be the most effective at the current time in the planning process [20]. As the space becomes over-sampled by simple samplers, more complex samplers will be able to take over. However, these samplers are applied globally over the whole problem, and the features of the planning space, such as topology, are not used when deciding where to apply the selected method.

Workspace-based Connectivity Oracle (WCO) [19] uses domain knowledge, i.e., workspace geometry and sampling history, to adaptively plan and sample the workspace. WCO is an adaptive sampling strategy that is composed of many component samplers each based on a geometric feature of a robot. By using adaptive hybrid sampling, it combines information from both workspace geometry and sampling history. This work improves the connectivity in narrow passages by generating milestones (nodes) that have a high probability of getting chosen. However, by using a K-closest approach (identifying k subset of closest neighbors from a set of all available nodes) once cannot guarantee that these nodes would be connectible. In [3], planning was also performed in workspace, uses a method previously used in image processing called the watershed method to help identify narrow passages in the workspace after which it adaptively samples the regions based on this retrieved information.

Learning-based Adaptive strategies Burns and Brock [4, 5] proposed an approach to multi-query motion planning using information from its previous experience to guide sampling to more relevant configurations. Every exploration of C-space provides information to a motion planner. They construct an approximate model of C-space. Their model captures and maintains information from each configuration and predicts the state of unobserved configurations to reduce collision detection calls.

RESAMPL [27] uses local region information (e.g., entropy of neighboring samples) to make decisions about both how and where to sample, and which samples to connect together. This use of spatial information about the planning space enables RESAMPL to increase sampling in regions identified as narrow and decrease sampling in regions identified as free. These approaches, briefly highlighted, use spatial information on where to apply planners and do not consider the topology that is discovered within the space that is being explored

The Unsupervised Adaptive Strategy (UAS) [31] is similar to feature sensitive motion planning in the sense that it identifies regions and specifies the planner to the region. UAS also considers the topology of the space. In UAS, K-means a clustering method, partitions the space using a training roadmap and applies hybrid PRM in each region. This method showed an improvement in speed and quality in the roadmaps generated, but does not consider all aspects of the planning process.

3 Adaptive Neighbor Connection via Node Characterization

Adaptive Neighbor Connection (ANC) [9] is a general connection framework that adaptively selects a neighbor finding strategy from a candidate set of options. ANC generates a set of candidate neighbors for a node q for PRM connection using a set of neighbor finders, NF $= nf_1, nf_2, \ldots, nf_m$. ANC learns a selection probability for each NF based on its prior success rate and cost, where success rate was the number of successful connections made per number of attempts and cost was determined by the number of collision detection calls used. It selects and updates probabilities similar to Hybrid PRM in [13]. ANC relieves the user the burden of hand selecting a connection method for each input problem which is challenging to do but did not consider roadmap quality versus actual computation cost in learning.

This paper improves on the ANC algorithm by enhancing the reward and cost function. ANC$_{nc}$ extends the reward function as shown in (Algorithm 1 and

Algorithm 2) of ANC to take into consideration the characteristics of the newly added nodes in the roadmap and the actual time spent to make the connection. Algorithm 1 makes use of the updated information being returned from Algorithm 2 which helps characterize the nodes and determines the better connector to use each time. A special distance metric well suited for articulated linkages was introduced in this paper called the knot theory distance metric. This metric looks into the similarity of configurations based on their topology and tangling patterns. ANC$_{nc}$ takes advantage of this distance metric in experiments we performed that looks at some higly articulated complex objects.

Algorithm 1 ANC

```
Input. A connecting vertex $q$, a set of neighbor finders $NF$, a local planner $lp$ and a
    graph $G$
Output. A connected graph $G$
Require: Let $P$ be a set of probabilities such that $p_i$ is the probability of selecting
    $nf_i$. Initialize $p_i = 1/|NF|$, $\forall p_i \in P$
 1: Randomly pick $nf_i$ according to $P$
 2: $N = nf_i$.FIND_NEIGHBORS$(q, G)$
 3: for each $n \neq q \in N$ do
 4:     if $lp$.IS_CONNECTABLE$(q, n)$ then
 5:         $G$.ADD_EDGE$(q, n)$
 6:     end if
 7: end for
 8: Let $r$ be the reward for roadmap improvement
 9: Let $c$ be the cost incurred
10: Update $(P, r, c)$
```

Algorithm 2 Update(P,r,c)

```
Input. A previous $prev$ and current state $curr$, where each state contains a connected
    component count $count$, time used $t$,local planner success $succ$ and local planner
    attempt $att$
Output. An adjusted reward based on node characterizaton, and cost based on time
    expended during the local planner connection attempt
1: if $curr.count \geq prev.count$ then
2:     $r = 1$
3: else
4:     $visibility = curr.succ/curr.att$
5:     $r = \epsilon^{-\gamma * visibility^2}$
6: end if
7: $c = curr.t$ - $prev.t$
8: Update P based on $r$ and $c$ in Equation 1 and 2 see [9]
```

3.1 ANC using Node Characterization

In [25], different characteristics were given to valid sampled configurations as they are added into the roadmap based on how they affect the connectivity and coverage of the roadmap during construction. Nodes were classified as:

- $cc - create$ if it forms a new connected component because it cannot be added to an existing one,
- $cc - merge$ if it connects to more than one connected component in the roadmap,
- $cc - expand$ if it connects to exactly one component in the roadmap and also increases the coverage of the roadmap see the visibility expansion criterion in [25], and
- $cc - oversample$ if it connects to exactly one connected component but does not increase the coverage of the roadmap.

We use this classification information in Algorithm 2 as our reward function and compare with the previous function described in [9]. By characterizing the types of nodes added and including information about their connectivity after each connection attempt, we gain insight about the type of nodes being added by each connector, and can better identify when a connector has reached its limit of usefulness i.e., it consistently connects oversampled nodes. Algorithm 2 takes this classification and rewards the connector as follows:

- reward nf_m maximally if the node that was added to the roadmap was a $cc - create$, $cc - merge$, otherwise
- reward nf_m based on the nodes visibility: $e^{-\mu v^2}$. Where v is the node's visibility.

A node q would be given a reward of 1 if after being connected to the roadmap, improved the both the coverage and connectivity. This occurs if node q merges with two or more existing connected component or expands the roadmap's coverage, by exploring areas that have not been reached beforehand. If q improves either coverage or connectivity or in some cases does nothing to improve the quality of the roadmap, then its reward becomes a function of how visible it is to other nodes in the roadmap which would be less than 1. In this way, a connector that consistently adds nodes that improves the roadmaps quality gains an increased knowledge about the C-space and would therefore have a higher probability of being used more often.

Difference between Reward and Cost of ANC and ANC$_{nc}$ In Algorithm 2, r is setup differently for ANC [9] and ANC$_{nc}$. In ANC, r is the reward given to the connectors as a results of the resulting success rate which is based on connection successes and failures. We make adjustments to this for ANC$_{nc}$ to take cognizance of the characteristics of the nodes that was just added to the roadmap by a particular connector based on criteria T.

The cost is also adjusted to record not only the number of collision calls by the local planner but also the actual time expended in making this connection. We calculate the cost as the difference in time between the previous connection attempt and the current attempt. This gives a more precise overview on the type of node that was added and how easy/difficult it was for the local planner to make the connection.

3.2 Knot Theory Distance Metric for Articulated Linkages:

In [10], knot theory was introduced as a similarity measurement. It looks into the topological similarity and differences of each configuration sampled. The concept of crossing numbers was used to determine when objects are topologically similar, and thus are better candidates as connection pairs.

Figure 1 gives a description of the crossing numbers. Projecting these edge vectors generates a parallelogram which is projected onto a unit sphere and the resulting area is calculated as the distance between the two links. This distance is

$$\delta_{\mathrm{KT}}(a, b) = 1/4\pi \sum_i \sum_j A_{ij}$$

where A_{ij} is the area on the sphere covered by vectors created between a and b.

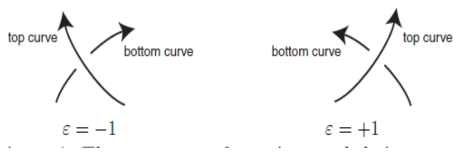


Fig. 1: Crossing numbers of two different edges.

We use this distance metric to help in identifying high DOF linkages that have formed knots within its configuration space. Such configuration are easily captured by this knot theory distance metric and thus return neighbors that are potentially connectable. It becomes more advantageous when using ANC$_{nc}$ because other distance measures are available that can quickly connect nodes without knots if they exist and also reduce the time to make this connections since this nodes added to the roadmap are closely monitored to see if they enhance the roadmap coverage and connectivity of not.

4 Experiments

We compare ANC$_{nc}$ to several other popular connection strategies and ANC. We first provide details on the experimental setup in Section 4.1. In Section 4.2, roadmaps are incrementally constructed until a query is solved for a variety of robots. Section 4.4 compares the performance of ANC$_{nc}$, ANC and a variation of ANC that includes the local planner time as the cost which is the same as we have for ANC$_{nc}$ but without the node characterization.

4.1 Experimental Setup

We implement all methods in the C++ motion planning library which uses the Standard Template Adaptive Parallel Library (STAPL), a C++ parallel library [6, 30]. We use RAPID [14] for collision detection.

We study the following environments (see Figure 2):

- **2D Heterogeneous.** (Figure 2(a)) This environment has 8 different rooms of different types including cluttered, free, and blocked that a stick-shaped robot must navigate.
- **2D Zig Zag.** (Figure 2(b)) A spherical robot must traverse a zig zag narrow passage.
- **2D S-Tunnel.** (Figure 2(c)) A cube robot must travel through an S shaped tunnel.
- **3D Walls.** (Figure 2(d)) The thin walls and openings produce situations where exact nearest-neighbor configurations will be difficult to connect for a stick robot.
- **3D Z-Tunnel.** (Figure 2(e)) The robot travels through a long narrow tunnel through the obstacle.
- **30 link insect like robot.** (Figure 2(f)) This robot has 3 link arms attached to a sphere each with 10dof and traversing an environment with different obstacles in the environment representing climbing and traversing scenarios.

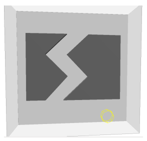
In our rigid body experiments we use obstacle-based sampling [2] and in our linkage experiments we use reachable volume sampling [23]. Connections are attempted between a node and its neighbors using a straight-line local planner. Neighbors are either defined as:

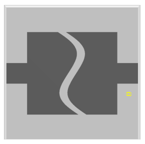
- K-Closest – the exact k nearest neighbors as given by some distance metric,
- K-Closest,K-Rand – k randomly selected neighbors from the exact k_2 nearest neighbors where $k_2 = 3k$ as in [22],
- R-Closest,K-Rand – k randomly selected neighbors from within a radius r, or
- K-Closest(Optimal) – k randomly selected neighbors that operate over two ranges of candidate neighbors and returns the k-closest pairs of the two ranges. Optimal is defined as closest pairs in this context.

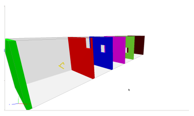
Distance metrics include scaled Euclidean with $s = 0.5$ (Euclidean) and center of mass (COM). ANC and ANC$_{nc}$ take all the above connection strategies as input.

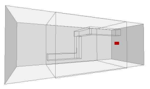
To examine performance, we analyze the time taken to solve an example query. We also gather statistics on the types of nodes connected by the different strategies.

(a) 2D Heterogeneous


(b) 2D Zig Zag


(c) 2D S-Tunnel


(d) 3D Walls


(e) 3D ZTunnel

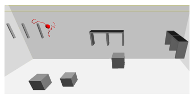

(f) 30 link insect like robot

Fig. 2: Problems studied. (a) A long rigid body robot in a heterogeneous 2D environment. A sample roadmap is shown. (b) A spherical ball in a 2D zig zag narrow passage. (c) A cube-like rigid body robot must navigate an S-tunnel in 2-D. (d) A 6 dof rigid robot must travel through a narrow hole in each wall. (e) An elongated cube must pass through a long narrow z-like tunnel. (f) A sphere with 3 long articulated arms attached to it.

4.2 Results

We compare the performance of each individual connection strategy, ANC [9], and ANC$_{nc}$ on each environment in Figure 2. All results are averaged over 5 runs. Table 1 shows the average time to solve the query and the different types of nodes created for each method. ANC$_{nc}$ is the fastest method in every environment except in 2D S-Tunnel where it is second to ANC.

It also needs the fewest nodes to solve the query. Figure 3(a) shows the learning plot for ANC$_{nc}$. This learning plot shows the probability trend and changes of all the connectors used by ANC$_{nc}$ in reference to the number of attempts. It learns K-Closest (Optimal) which is not the fastest method but uses the least nodes. ANC, on the other hand, learns a different connection method, K-Closest(COM). (see Figure 3(b)).

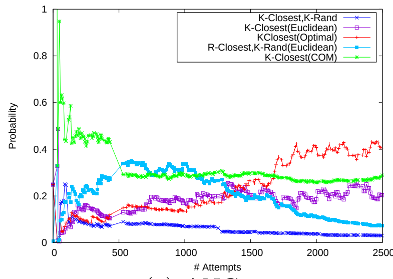
In the 2D Zig Zag environment, ANC$_{nc}$ also performs best in terms of time. We see from Figure 4(a) that it does so by learning the best performing individual connection method (K-Closest,K-Rand) in terms of time and number of oversampled nodes. ANC instead learns the K-Closest(Euclidean) method (see Figure 4(b)). This aids it in outperforming individual connection methods but not in outperforming ANC$_{nc}$.

The 2D S-tunnel environment shows a case where ANC$_{nc}$ is outperformed minimally by ANC. Interestingly, they learn different overall connection methods and at different learning rates (see Figure 5). While it takes marginally longer, ANC$_{nc}$ needs fewer merge(good) nodes than ANC to solve the query.

In the 3D environments, ANC$_{nc}$ outperforms all other methods in terms of time, uses fewer nodes to solve the query, and minimizes the effects of expanded

Table 1: Each method constructs a roadmap until the query is solved. ANC and ANC$_{nc}$ are comprised of the other 5 connection methods. All results are averaged over 5 runs. Boldface entries indicate the most desirable time.

Environment	Method	Merge	Create	Expand	Oversample	Time (s)
2D Heterogeneous	K-Closest,K-Rand	276	9	209	0	0.242
	K-Closest(Euclidean)	182	6	6	5	0.389
	K-Closest(Optimal)	73	3	99	21	0.612
	R-Closest,K-Rand	96	4	0	144	0.636
	K-Closest(COM)	182	6	6	149	0.388
	ANC	220	5	311	0	0.484
	ANC$_{nc}$	59	2	29	55	**0.163**
2D Zig-Zag	K-Closest,K-Rand	301	8	4265	0	5.52
	K-Closest(Euclidean)	698	9	0	6731	6.91
	K-Closest(Optimal)	698	3	1805	3632	6.84
	R-Closest,K-Rand	698	9	0	6731	6.93
	K-Closest(COM)	249	6	0	408	6.96
	ANC	228	6	0	427	6.08
	ANC$_{nc}$	55	1	0	160	**0.35**
2D S-Tunnel	K-Closest,K-Rand	18964	92	434671	0	420
	K-Closest(Euclidean)	9437	20	0	85963	256
	K-Closest(Optimal)	9435	20	0	423803	634
	R-Closest,K-Rand	12273	60	235706	134	400
	K-Closest(COM)	9437	20	0	85963	167
	ANC	4650	16	0	3767	**105**
	ANC$_{nc}$	2311	16	0	6205	114
3D Walls	K-Closest,K-Rand	30146	31	322235	4	720
	K-Closest(Euclidean)	12002	7	0	108131	691
	K-Closest(Optimal)	2001	2	0	82081	137
	R-Closest,K-Rand	45199	1	0	895000	950
	K-Closest(COM)	12002	7	0	108131	689
	ANC	5174	23	1198	11602	229
	ANC$_{nc}$	36	0	0	1962	**84**
3D Z-Tunnel	K-Closest,K-Rand	87268	102	1926922	0	22400
	K-Closest(Euclidean)	21997	13	0	200210	3247
	K-Closest(Optimal)	21995	12	0	1100121	18257
	R-Closest,K-Rand	3964	3	296	1552649	20198
	K-Closest(COM)	21997	13	0	200210	3256
	ANC	21353	13	0	19055	3400
	ANC$_{nc}$	309	3	1	7685	**2386**


(a) ANC$_{nc}$

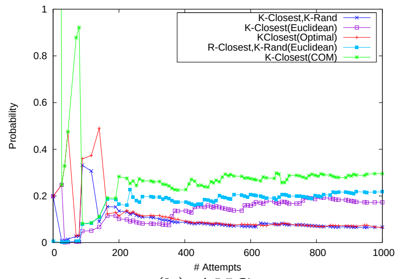
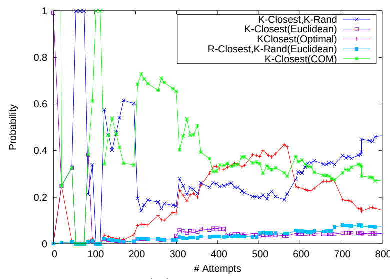

(b) ANC

Fig. 3: Learning plots for the 2D Heterogeneous environment.


(a) ANC$_{nc}$

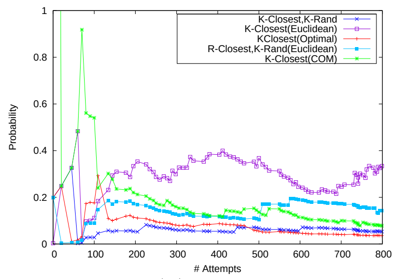

(b) ANC

Fig. 4: Learning plots for the 2D Zig Zag environment.

or oversampled nodes. Figure 6(a) shows a rapid learning curve for ANC$_{nc}$ as it quickly identifies the connection strategy that produces the most desirable types of nodes to aid in solving the query. Figure 6(b) shows a slower learning curve for ANC. Even though ANC$_{nc}$ did not choose the best individual connection strategy while ANC did, it was still able to produce a query-solving roadmap the fastest, largely due to its small number of oversampled nodes.

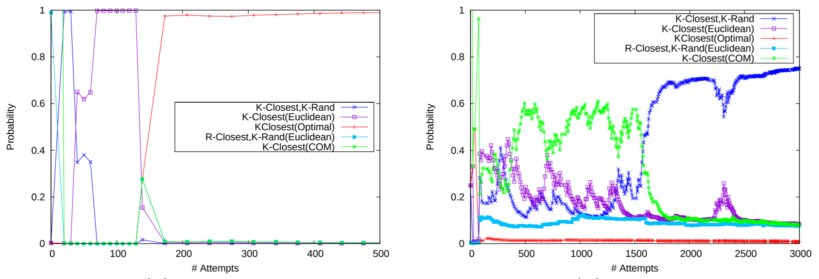
Table 2 compares the running time savings of ANC$_{nc}$ over ANC and over the best individual component method for each environment. ANC$_{nc}$ is faster in all cases except one and requires on average only half the time of either competitor.

4.3 Complex Environment Scenario

The results in Table 3 shows the performance of ANC$_{nc}$ and shows an improvement in the connection time where the next shortest time take double the time.

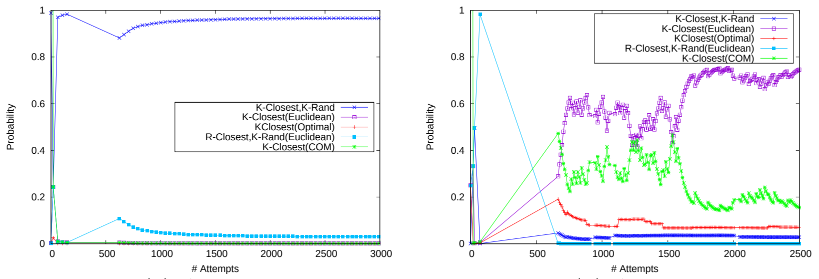
4.4 The Importance of Node Characterization

Recall that ANC$_{nc}$ differs from ANC in two main ways: it uses node characterization to assign rewards and it uses the actual connection time (and not the



(a) ANC$_{nc}$ (b) ANC

Fig. 5: Learning plots for the 2D S-Tunnel environment.



(a) ANC$_{nc}$ (b) ANC

Fig. 6: Learning plots for the 3D Z-Tunnel.

number of collision detection calls) to assign cost. To isolate out the benefit of node characterization, we compare ANC$_{nc}$ to two versions of ANC: ANC (as published in [9]) and ANC$_t$ where cost is connection time instead of the number of collision detection calls. Thus, the only difference between ANC$_{nc}$ and ANC$_t$ is how rewards are computed, and the only difference between ANC$_t$ and ANC is how costs are computed. Figure 7 compares the running times for each method in all the environments. We see that while including the actual time in the cost improves performance (ANC$_t$ versus ANC), it alone does not account for the improved performance of ANC$_{nc}$ over ANC.

5 Conclusion

Choosing the appropriate connection method for a given input problem is crucial to performance and challenging to do. We presented an intelligent way of selecting connection strategies for PRM roadmaps that uses the types of nodes created to reward strategies and the time required to create connections to penalize strategies. The resulting method, ANC$_{nc}$, outperforms individual connection strategies as well as previous work in a variety of test cases save one. It is able to solve the query faster than other methods and in many cases with smaller

Table 2: Running time comparison of ANC$_{nc}$ verses ANC and the best individual component method for each environment.

Environment	ANC$_{nc}$/ANC	ANC$_{nc}$/Best_Individual
2D Heterogeneous	0.337	0.674
2D Zig Zag	0.058	0.063
2D S-Tunnel	1.086	0.683
3D Walls	0.367	0.613
2D Z-Tunnel	0.702	0.735
Average	0.510	0.554

Table 3: Running time comparison of ANC$_{nc}$ verses ANC and the best individual component method for each environment.

ConnectType	NumNodes	NumEdges	Numconnec	CClargest	QueryTime	ConnectionTime
Adapt	2006	8428	226	1775	52.90	229.552
RRand	4505	17738	539	3947	128.323	440.919
S.euclidean	509	43668	5	505	42	2720.43
S.eu/knot	2007	4780	573	1434	735.251	1361.06
Optimal	13492	345402	274	13219	139.529	9499.73

roadmaps. Future direction to this work involves varying the local planners that are being used to give a sense of the different time that is being expended and adding this as one of the learning factors.

References

1. R. Alterovitz, M. Branicky, and K. Goldberg. Motion planning under uncertainty for image-guided medical needle steering. *Int. J. Robotics Research*, pages 1361–1374, 2008.
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 the third workshop on*

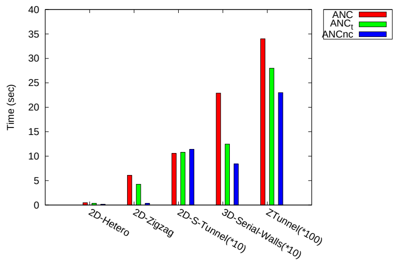


Fig. 7: Running times of ANC, ANC$_t$ and ANC$_{nc}$ for each environment.

the algorithmic foundations of robotics on Robotics : the algorithmic perspective: the algorithmic perspective, WAFR '98, pages 155–168, Natick, MA, USA, 1998. A. K. Peters, Ltd.

3. J. Berg and M. Overmars. Using workspace information as a guid to non-uniform sampling in probabilistic roadmap planners. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 453–460, 2004.
4. B. Burns and O. Brock. Sampling-based motion planning using predictive models. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3313–3318, 2005.
5. B. Burns and O. Brock. Toward optimal configuration space sampling. In *Proc. Robotics: Sci. Sys. (RSS)*, pages 105–112, 2005.
6. A. Buss, Harshvardhan, I. Papadopoulos, O. Pearce, T. Smith, G. Tanase, N. Thomas, X. Xu, M. Bianco, N. M. Amato, and L. Rauchwerger. STAPL: Standard template adaptive parallel library. In *Proc. Annual Haifa Experimental Systems Conference (SYSTOR)*, pages 1–10, New York, NY, USA, 2010. ACM.
7. J. F. Canny. *The Complexity of Robot Motion Planning.* MIT Press, Cambridge, MA, 1988.
8. A. Davids. Urban search and rescue robots: From tragedy to technology. *IEEE Intelligent Systems*, 17(2):81–83, 2002.
9. C. Ekenna, S. A. Jacobs, S. Thomas, and N. M. Amato. Adaptive neighbor connection for prms: A natural fit for heterogeneous environments and parallelism. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 1–8, November 2013.
10. M. A. Erdmann. Protein similarity from knot theory and geometric convolution. In *Proceedings of the eighth annual international conference on Resaerch in computational molecular biology*, RECOMB '04, pages 195–204, New York, NY, USA, 2004. ACM.
11. R. Geraerts and M. H. Overmars. Reachablility analysis of sampling based planners. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 406–412, 2005.
12. J. P. Hespanha, H. J. Kim, and S. Sastry. Multiple-agent probabilistic pursuit-evasion games. In *Proceedings of the IEEE Conference on Decision and Control*, pages 2432–2437, 1999.
13. D. Hsu, G. Sánchez-Ante, and Z. Sun. Hybrid PRM sampling with a cost-sensitive adaptive strategy. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3885–3891, 2005.
14. D. Johnson and E. Cohen. A framework for efficient minimum distance computations. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 4, pages 678–3684, 1998.
15. L. E. Kavraki, J.-C. Latombe, R. Motwani, and P. Raghavan. Randomized query processing in robot path planning. In *Proc. ACM Symp. Theory of Computing (STOC)*, pages 353–362, May 1995.
16. L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.
17. H. Kitano, S. Tadokoro, I. Noda, H. Matsubara, T. Takahashi, A. Shinjou, and S. Shimada. Robocup rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. In *IEEE Trans. Sys., Man, Cybern.*, 1999.
18. J. J. Kuffner and S. M. LaValle. RRT-connect: An efficient approach to single-query path planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 995–1001, 2000.
19. H. Kurniawati and D. Hsu. Workspace-based connectivity oracle - an adaptive sampling strategy for prm planning. In *Algorithmic Foundation of Robotics VII*,

pages 35–51. Springer, Berlin/Heidelberg, 2008. book contains the proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR), New York City, 2006.

20. S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *Int. J. Robot. Res.*, 20(5):378–400, May 2001.
21. T. Lozano-Perez. Spatial planning: A configuration space approach. *Computers, IEEE Transactions on*, 100(2):108–120, 1983.
22. T. McMahon, S. Jacobs, B. Boyd, L. Tapia, and N. Amato. Evaluation of the k-closest neighbor selection strategy for prm construction. Technical Report TR12-002, Texas A&M, College Station Tx., 2011.
23. T. McMahon, S. Thomas, and N. M. Amato. Sampling based motion planning with reachable volumes: Theoretical foundations. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2014.
24. M. Morales, L. Tapia, R. Pearce, S. Rodriguez, and N. M. Amato. A machine learning approach for feature-sensitive motion planning. In *Algorithmic Foundations of Robotics VI*, pages 361–376. Springer, Berlin/Heidelberg, 2005. book contains the proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR), Utrecht/Zeist, The Netherlands, 2004.
25. M. A. Morales A., R. Pearce, and N. M. Amato. Metrics for analyzing the evolution of C-Space models. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1268–1273, May 2006.
26. E. Plaku and L. Kavraki. Quantitative analysis of nearest-neighbors search in high-dimensional sampling-based motion planning. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, July 2006.
27. S. Rodriguez, S. Thomas, R. Pearce, and N. M. Amato. (RESAMPL): A region-sensitive adaptive motion planner. In *Algorithmic Foundation of Robotics VII*, pages 285–300. Springer, Berlin/Heidelberg, 2008. book contains the proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR), New York City, 2006.
28. G. Song and N. M. Amato. A motion planning approach to folding: From paper craft to protein folding. *IEEE Trans. Robot. Automat.*, 20:60–71, February 2004. Preliminary version appeared in *ICRA 2001*, pp. 948-953.
29. G. Song, S. L. Thomas, and N. M. Amato. A general framework for prm motion planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4445–4450, 2003.
30. G. Tanase, A. Buss, A. Fidel, Harshvardhan, I. Papadopoulos, O. Pearce, T. Smith, N. Thomas, X. Xu, N. Mourad, J. Vu, M. Bianco, N. M. Amato, and L. Rauchwerger. The STAPL Parallel Container Framework. In *Proc. ACM SIGPLAN Symp. Prin. Prac. Par. Prog. (PPoPP)*, pages 235–246, San Antonio, Texas, USA, 2011.
31. L. Tapia, S. Thomas, B. Boyd, and N. M. Amato. An unsupervised adaptive strategy for constructing probabilistic roadmaps. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4037–4044, May 2009.
32. R. Vidal, O. Shakernia, H. Kim, D. H. Shim, and S. Sastry. Probabilistic pursuit-evasion games: Theory, implementation, and experimental evaluation. In *IEEE Transaction son Robotics and Automation*, pages 662–669, 2002.