Bias-guided Metrics for Motion Planning Problems

Regina Rex[1], Bonnie Wang[2], Diane Uwacu[3], Shawna Thomas[3], Nancy M. Amato[4]*

[1] Department of Mathematics and Computer Science,
University of Wisconsin-Superior, Superior, WI, USA,
rrex@uwsuper.edu
[2] Department of Computer Science,
Columbia University, New York, NY, USA,
bw2551@columbia.edu
[3] Parasol Lab, Department of Computer Science and Engineering,
Texas A&M University, College Station, TX, USA,
{duwacu, sthomas}@tamu.edu
[4] Parasol Lab, Department of Computer Science,
University of Illinois at Urbana-Champaign, IL, USA,
namato@illinois.edu

Abstract. Despite using current motion planning algorithms to investigate and find paths within a workspace, it is difficult and inefficient to evaluate vastly complex and different environments with a standard planning strategy. We introduce a biasing planning strategy, using specific metrics tailored to the environment in order to discover the most desirable paths faster. In regards to protein environments, we use motion planning algorithms to evaluate and understand the accessibility of a drug molecule (ligand) to a binding site within a protein. To better predict the ligands path to the site, we bias our planning strategy towards lower energy pathways by annotating our protein model with biometrics, such as energy or clearance, resulting in a more informed and accurate model of the accessibility routes.

1 Introduction

Motion planning refers to the process of finding an obstacle-free path for robots, given a starting point and a goal destination. Some applications of motion planning includes computational biology, prototyping and graphics [1]. A variety of problems involving a "robot" subject and a target can be formulated as a motion planning problem.

Although these problems can be boiled down to a geometric problem, it is computationally hard to plan for robot navigation because of how difficult it is to capture all the robot constraints and represent the environment as a simple model to a computer.

* This work was performed at the Parasol Lab-UIUC during Summer 2019.

There has been much research done on creating faster and more complex planners. Currently, existing motion planning methods use the workspace property to guide planning process. To map out the workspace, the free space, or areas of valid configurations, is constructed from the environment's properties.

Narrow passageways and complex environments has always been a persistent dilemma for motion planning algorithms, and there has been many different approaches to build an algorithm robust enough. One such strategy is guided motion planning. A skeleton based on the environment's topology is first created before the planning phase begins. The planner will then build its roadmap based on the regions represented by the skeleton.

Still, even with the use of state-of-the-art skeleton-guided planners, it is difficult to explore specific regions in a complex environment. The protein's topology alone is not enough information to efficiently explore the space. In addition, while looking at the applications where guided motion planning would be useful, not only does finding a path matter, but the quality of the path matters as well. For example, in a robotics environment, where you want the robot to traverse the safest space, you would want the paths of the highest clearance. In a protein environment, where you would predict the protein structure to remain in its most stable configuration, you would want the paths of the lowest energy.

Our Contribution: We introduce the idea of using metrics for biasing workspace exploration. More specifically, we use workspace properties such as clearance and energy to guide the planning process for the robotics and protein application of motion planning. Both metrics are useful in problems which have a strong correlation to their workspace geometry.

Robotics (Clearance): In robotics, we use clearance to guide how a robot explores its environment. Using the clearance metric, we can find safer paths in a faster time, while also addressing the bottlenecks of the state-of-the-art planning method (sampling-based planning).

Protein-Ligand Binding (Energy): We can extend this method to the Protein-Ligand Binding problem in computational biology. Ligands are small drug molecules that when interacted with proteins, change their shape and functionality. These interaction regions are known as binding sites and can be located within the protein's inner hull. We can study the protein as a geometric and energetic problem, applying both the clearance biasing method as well as an additional energy metric biasing method. These metrics can help gain insight on how a ligand navigates around the tunnels of the protein. Only recently, protein-ligand binding methods began considering the feasible paths a ligand can take to the binding site. Previous work mainly focused on the final fit of the ligand to the binding site, without considering the path the ligand must traverse to reach its goal. Current work studies these possible paths, investigating how the protein's tunnels can regulate the accessibility of certain ligands to the site [3].

Our method strives to understand and evaluate the tunnels of the protein that the ligand would most likely be able to access in order to reach the binding site. Not only is accessibility of a tunnel important, but probability of a tunnel being a true path a ligand could take should also be taken into account. In order to gain insight on meaningful tunnels, we want to find tunnels of high volume and low energy for which the protein-ligand complex will increase in stability [3]. With our biasing planning strategy, we can narrow our search of our environment to a certain criteria, and thus investigate and hone in on meaningful tunnels faster. Using a comparison experiment between a non-biasing planning strategy and two biasing strategies (clearance and energy), we can show improvements in efficiency in how we search the environments.

Our experiments show that using our metrics to bias our planning, we are able to generate desirable paths faster for both the robotics and protein application of motion planning.

2 Related Work

In this section, we would discuss the preliminaries of our work.

2.1 The Motion Planning Problem

Motion Planning is also known as the geometry path planning problem. A configuration represents a complete specification of the position of every point on the robot. And the set of all the possible configurations of the robot is known as the Configuration Space C_{Space}. The C_{Space} is made up of Obstacle Space C_{Obst} and Free Space C_{Free}. C_{Obst} is the set of all configurations which lies in one or more obstacles, and C_{Free} represents the set of all configurations that are not in C_{Obst}.

The configuration space is an abstract model of the physical environment of the robot and obstacles, know as workspace. As such, this model does not capture all the possible constraints of the workspace. The robot is represented as a configuration in C_{Space}.

The motion planning problem is defined in C_{Space} as follows: Given a start configuration ($q_s \in C_{Free}$) and a goal configuration ($q_g \in C_{Free}$). Return a continuous path, $p : [0, 1] \rightarrow C_{Free}$ such that $p(0) = q_s$ and $p(1) = q_g$.

Other non-robotics applications can be formulated as a motion planning problem, if they can be map to the geometric definition given above. For example, the study of Protein-Ligand interaction is a motion planning problem, where the robot is the ligand, and the geometric representation of the protein is the workspace.

Protein-Ligand Binding Problem A protein is a large structure made up of a chain of amino acids that interacts in several essential reactions in the body [3]. These reactions can include binding with a small drug molecule called a ligand. The region the ligand interacts with on the protein is known as the binding site. Once the interaction occurs, the protein's shape and functionality can change.

Recent research indicates that the molecular tunnels of the protein regulate the accessibility of ligands and is dependent on those ligands' specific characteristics. These tunnels show a connection with how binding site activity behaves.

Investigating how the ligand is able to travel to the binding site can provide insight in how protein-ligand binding works and how to predict certain biological phenomena.

2.2 Sampling Based Planning

This is one of the approach to solving the motion planning problem. This approach avoids creating an explicit construction of C_{Obst}, instead they treat C_{Space} as a blackbox, and randomly sample configurations in C_{Space} using a collision detection check to see if it's valid, and builds a roadmap, which is then searched for a valid path. a roadmap is a topological graph, which ever vertex represents a configuration, and each edge is a path connecting two configurations. Collision Detection Check is a black box which defines if a configuration is value or not. Collision Detection Checks are expensive because they involve translating the problem between C_{Space} and workspace. Another bottle neck for this approach is narrow passages. Narrow passages-parts of the environment where the probability of sampling a valid object position is low. Sampling-based planners have a hard time finding narrow-passages.

Rapidly-exploring Random Trees (RRTs) A Rapidly-exploring Random Tree (RRT) is a sampling-based planner which takes a tree-based approach to solving the motion planning problem. Tree-based planners are planners that builds a tree roadmap during planning. At each extension attempt of an RRT, it generates a random configuration, q_{rand} which is connected to the nearest configuration, q_{near}, If there no obstacles exist between q_{rand} and q_{near}. RRTs are good for solving single query problems and they tend to explore the workspace better than other sampling-based planners.

Probabilistic Roadmap (PRM) PRM takes a graph based approach to solving the motion planning problem [2]. This planning strategy creates a graph or roadmap by repeatedly generating random possible configurations and then attempting to connect them. As a result, the graph that is created will be entirely in free space since the planner will check for collisions when connecting the roadmap.

PRM has an advantage over other planning strategies because it is able to handle multiple queries. In other words, PRM can search and find multiple paths with a single generated roadmap.

2.3 Guided Motion Planning

Topological Guidance Topological guidance involves using the workspace structure to direct how sampling based planners like PRM and RRT explores the environment.

Some motion planning approach to the motion planning problem utilizes the workspace decomposition [] for planning, specifically for targeting narrow passages. The workspace decomposition involves partitioning the workspace into tetrahedral which are used to bias the sampling process. Such an approach focuses on the narrow passage bottlenecks of motion planning, but they result in oversampling in some regions as opposed to others using these approach.

Another approach is the used of Skeletons made from the workspace decomposition to bias sampling in C_{space} []. Workspace Skeletons are graphs which captures the topological features of the environment. The Bias-guided method relies on topological guidance sampling-based planners. The goals of our method is to exploit the topological properties of such methods using metrics that are beneficial to specific workspace corrolated problems. As mentioned in the introduction, our method is applied to DR-RRT and DR-RRG. For both skeleton-guided planners, we use Mean Curvature Skeleton (MCS) for generating our workspace skeleton [7].

MCS constructs the workspace skeleton using a mesh-based algorithm to compute its skeletal representation from the mean curvature flow of surfaces in the workspace.

Planning We use different planners to build our roadmap based on the motion planning problem. Like mentioned before, there are different sampling planning methods that would have advantages in certain environments over others. We can augment the previously discussed sampling based planning methods in section 2.2 with a dynamic region-biased strategy [7].

Dynamic region-biased strategies guide the planner to sample only from particular regions based on the workspace topology. A skeleton, or graph that maps the essence of the free workspace topology, will be created to direct where to select the regions from. These regions will then be created, sampled from and then destroyed as the planner explores narrow and complex passageways.

Dynamic Region Rapidly-exploring Random Tree (DR-RRT) is a skeleton-guided RRT which uses the workspace skeleton to bias RRT growth. In other words, the planner will grow the tree from samples generated inside the selected region and will advance to the next region once those regions have been represented. DR-RRT is faster and returns lower collision detection calls when compared to basic RRT.

Dynamic Region Rapidly-exploring Random Graph (DR-RRG) is a skeleton-guided strategy that combines PRM and RRT [8]. Like the DR-RRT strategy, the planner will choose a region to sample from using RRT; however, it will then attempt to connect these samples to form a roadmap to increase connectivity. DR-RRG has advantages in increased efficiency with multiple queries.

3 Method

The Bias-guided method is designed for skeleton-guided RRTs and applied to DR-RRT for the robotics application and DR-RRG for the protein application.

In this section, we discuss how the algorithm works and give a detail example of its planning process.

3.1 Algorithm Overview

Our method is designed for skeleton-guided RRTs and applied to DR-RRT or DR-RRG for the robotics application and DR-RRG for the protein application. The dynamic region method utilizes workspace properties such as clearance and energy to guide how skeleton-guided planners generate roadmaps. Algorithm 1 details a general dynamic region planning strategy, called DR-RDMP for simplicity in our explanation. This general algorithm will be slightly modified in regards to DR-RRT and DR-RRG in how the planner will grow its roadmap (line 8).

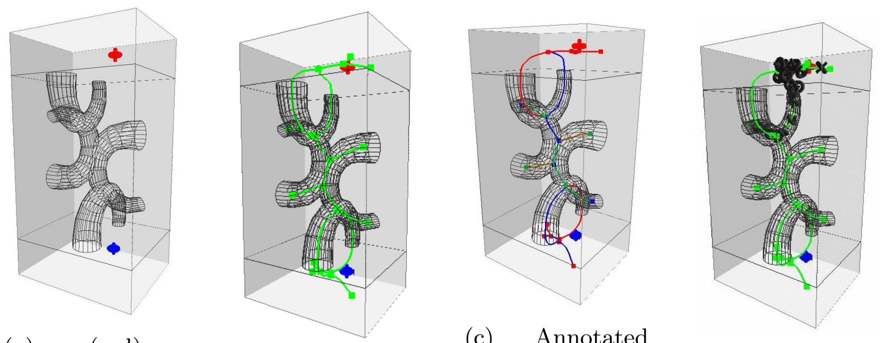
Algorithm 1 Bias-guided DR-RDMP

```
Input: Environment $env$, Start $s$, Goal $g$,
    Bias Metric (min/max, clearance/energy) $biasMetric$
Output: Path $p$
 1: $WS \leftarrow$ GetWorkspaceSkeleton($env$)
 2: $AS \leftarrow$ AnnotateSkeleton($WS$)
 3: $g \leftarrow s$
 4: $r \leftarrow$ GetInitialRegion($WS, s$)
 5: while $\neg done$ do
 6:     $C_r \leftarrow$ GetChildren($r$)
 7:     $r \leftarrow$ SelectRegion($C_r$, $biasMetric$)
 8:     $R \leftarrow$ GrowRDMP($r$)
 9: end while
10: $p \leftarrow$ Query($R, g$)
11: return $p$
```

In order to go in depth with our algorithm, we will be using the 3D MazeTunnel environment (Figure 1) as an example.

Generally, our method will take in the environment, a start and a goal query as input. In addition, it will also need the bias metric, energy or clearance, and if the planner should favor the maximum or minimum values of those metrics.

With the input of the environment, the algorithm will be able to decompose the areas of free space into geometric tetrahedral shapes. With this, we will able to generate our workspace skeleton described in section 2.3. In Figure (1b), you can see that the skeleton, colored in green, is able to cover and represent the major regions of free space around and inside the MazeTunnel. For the generated workspace skeleton, we annotate it with properties such as clearance and energy. The resulting annotated skeleton can be seen in Figure (1c), which is augmented with clearance values. The Annotated Skeleton colors the highest clearance regions starting with red, and then to yellow with the lowest clearance being green and then blue.



(a) q_s (red), q_g (blue) (b) Skeleton (c) Annotated skeleton (d) Planning

Fig. 1: Example execution of the Bias-guided method: (a) env with q_s and q_g; (b) MazeTunnel Workspace Skeleton (green) (c) Workspace Skeleton annotated with clearance value (red highest clearance, blue low clearance) (d) Minimum clearance-bias exploration process (half-way).

Using this skeleton, we direct the planning process based on the annotated properties. To begin building our roadmap, the start of the roadmap will be placed at the goal configuration. This will ensure that the algorithm can explore as many paths or tunnels that will lead to the goal (line 3). Our algorithm will then build its roadmap by selecting a region based on the biased annotations of the skeleton, sample from that region and then grow the roadmap (line 5-8). In Figure (1d), roadmap construction is guided towards regions with minimum clearance value.

How the algorithm grows its roadmap varies slightly between DR-RRT and DR-RRG. In line 8, DR-RRT will extend its roadmap by connecting the new samples to the original connected tree. DR-RRG will instead extend its roadmap by creating smaller graphs from connecting the k closest samples to one another.

After the roadmap is successfully built, this concludes the guided planning step. The query phase can be performed after to find the paths to the goal (line 10).

3.2 Metrics

This section covers detailed description of the metrics used for the Bias-guided method.

Computing Clearance Clearance is defined as the size of free space between obstacles in the environment. Given a workspace skeleton, we compute clearance using the skeleton node and the point closest to the nearest obstacle. The distance between this point and the skeleton node is what we define as its clearance value.

To annotate the workspace skeleton with its clearance value, we do the following: Calculate the clearance value of every node in the workspace skeleton. Then, the edge clearance value is assigned as the maximum or minimum node clearance value, which makes up the edge. We pick the maximum node clearance for minimum Bias-guidance and minimum node clearance for maximum Bias-guidance. This way, we ensure that we select the best option available for each step of the planning process.

Computing Energy-value Our energy function calculates the energy of each region based on Van der waals interactions, which are weak intermolecular forces between two atoms or molecules [9]. How strong the attraction between two atoms are determined by the distance they are from one another; as the atoms move closer the energy of the system decreases, but if they come too close the atoms start to repel each other and dramatically increase the energy. The energy values the function outputs for is determined by first discretizing the workspace into grid cells and determining the influence of the force of every atom of the protein on the ligand.

The workspace skeleton energy annotation is defined as follows: For each edge in the workspace skeleton, we calculate the energy of its nodes. We assign the highest node energy value as the energy value of the edge for minimum energy-bias, and the lowest node energy value for maximum energy-bias.

3.3 Visualization Tools

In the same way the skeletons are visualized in Figure (1b), we needed to help visualize the biasing effect of our new strategy. Our current motion planning visualization tool called Vizmo++ is useful in being able to interact and visualize with skeletons, environments and roadmaps. In order to test how well our biasing strategy will influence the construction of the roadmap, we decided to develop a feature that is able to color the skeleton with the annotation values. The feature is compatible with both clearance and energy annotations, and can be adaptable to show any given amount of intervals in colors. In Figure (1c), it pictures our annotated skeleton with red being areas of highest clearance and blue being areas of lowest. The feature is extremely useful not only for debugging purposes, but showing the advantages of using the biasing strategy.

4 Experiments

We tested our method in robotics and protein environments. The Bias-guided method is implemented in C++ using the Parasol Motion Planning Library (PMPL) and Standard Template Adaptive Parallel Library (STAPL); both libraries are developed in the Parasol Lab. All the experiments were run on a 3.40GHz Intel Core i7-3770 CPU.

We use the motion planning visualization tool, Vizmo++. Vizmo++ is developed at the Parasol Labs. And it is used for viewing and editing motion

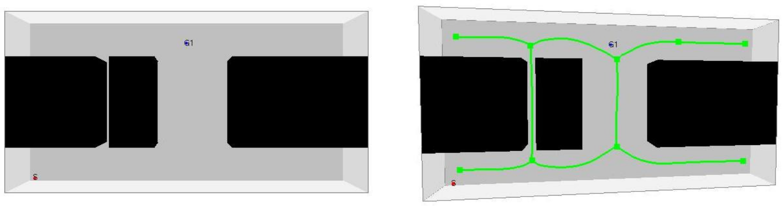

(a) Obstacles with q_s, q_g (b) Obstacles with MCS

Fig. 2: Obstacles robotics environment

planning problems and their resulting roadmaps, or solution. For our experiments, we use Vizmo++ to visualize the robotics and protein workspace, and the solution returned from the strategies.

Environment	Strategy	Runtime	CDC	Nodes	Edges	Path Clearance
Obstacles	Regular DR-RRT	0.33	87,841	267	531	5.05
	Max Clearance-bias	**0.18**	**40,510**	**131**	**260**	**6.12**
	Min Clearance-bias	0.22	52,287	177	353	3.40
MazeTunnel	Regular DR-RRT	0.46	11,553	98	194	**0.93**
	Max Clearance-bias	**0.18**	**6,158**	**65**	**127**	0.92
	Min Clearance-bias	1.45	28,077	88	173	0.85

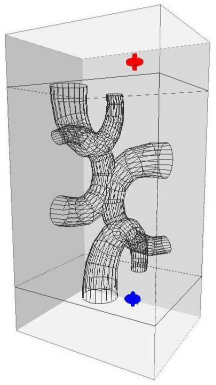
Table 1: Robotics experiment results. Bolded values indicate the best value for each environment. The best value for Runtime, CDC, Nodes, and Edges is the lowest number in the column, while the best value for the Path Clearance is the highest number, which indicates the safest path.

4.1 Robotics Experiments

Environment Setup In the robotics experiments, we compare our method to the regular DR-RRT. We used DR-RRT because it is a skeleton-guided RRT that utilizes the workspace decomposition for directing RRT growth.

We ran our experiment in MazeTunnel (Figure 3a), and Obstacles (Figure 2a) using holonomic robots. Both environments have wide and narrow passage options. We selected these 3D environments to demonstrate the process of our method and its advantages in such situations. For the workspace skeleton, we use the Mean Curvature Skeleton (MCS). MCS provides a better workspace skeleton for environments with convex bodies like the MazeTunnel environment, but the skeleton quality is not ideal for environments with complex concave bodies.

Each experiment ran until the query was solved. We performed trials using ten random seeds for each strategy in both environments, and we averaged


(a) MazeTunnel with q_s, q_g

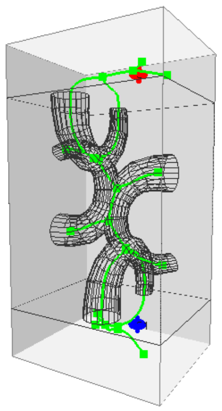
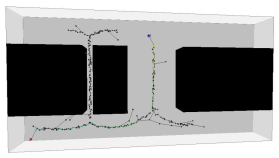
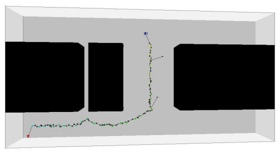

(b) MCS for MazeTunnel

Fig. 3: MazeTunnel robotics environment

the following over all the trials: runtime, nodes, edges, collision detection calls (CDC), and path clearance.


(a) Regular DR-RRT


(b) Max Clearance-bias

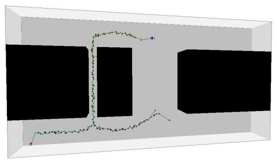

(c) Min Clearance-bias

Fig. 4: The Roadmap and path from running the different strategies in Obstacles.

Result: Table 1 contains the data collection from the robotics experiment. Runtime is the time (in seconds) taken for each strategy to find a path. CDC is the amount of collision detection checks performed while the strategy runs. Nodes and Edges refer to the number of nodes and edges in the resulting roadmap. And Path Clearance is the clearance value of the path return by the strategy. Runtime, CDC, Nodes, Edges, and Path Clearance are averaged over all the ten trials performed. Figure 4 and 5 shows the roadmap (in black) and the resulting path (highlighted) from running the experiments in Obstacles and MazeTunnel environment.

Discussion: In Obstacles, the results table 1 notes that both minimum and maximum clearance-bias have lower CDC than regular DR-RRT. The resulting

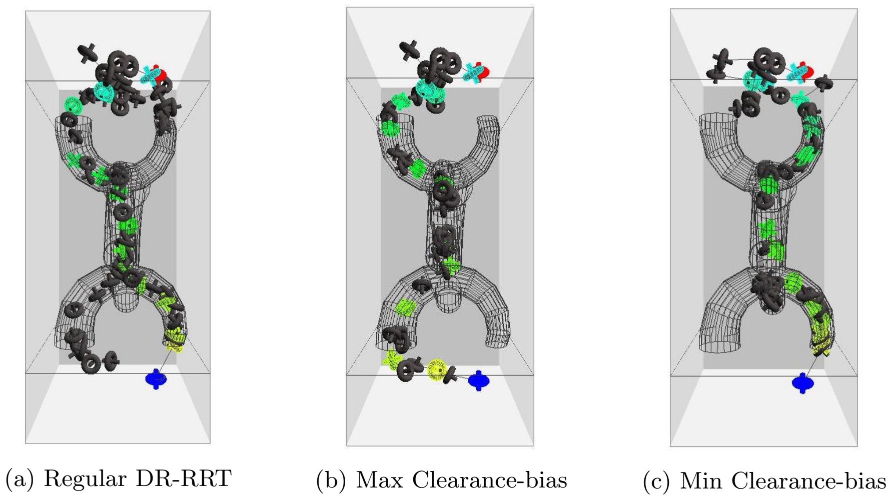

(a) Regular DR-RRT (b) Max Clearance-bias (c) Min Clearance-bias

Fig. 5: The Roadmap and Path from running the different strategies in MazeTunnel.

roadmaps with these strategies also have fewer nodes and edges when compared to regular DR-RRT. Minimum clearance-bias returns the path with the lowest clearance value, while maximum clearance-bias returns the path with the widest clearance. The Bias-guided method also returns the safest path in the quickest time using maximum clearance-bias.

Maximum clearance-bias returns the widest path in MazeTunnel with lower CDC, nodes, and edges when compared to regular DR-RRT. And it is the fastest strategy in MazeTunnel. Minimum clearance-bias was the slowest, but it had fewer nodes and edges than regular DR-RRT.

From our experiments, we found that biasing roadmap construction using a workspace property such as clearance, help speed up the planning process in robotics environments like Obstacles and MazeTunnel.

4.2 Protein Experiments

Environment Setup For the protein experiments, we conduct a comparison experiment between the energy-bias method, the clearance-bias method and a non-biased planning method. We biased towards minimum energy for the energy-bias method and maximum clearance for the clearance-bias method. As mentioned in the related work, the non-biased planning method follows the same guided planning, but chooses regions to sample from based on the protein's topology rather than energy or clearance. For these complex protein environments, the planner we use for all three methods is a Mean Curvature Skeleton guided DR-RRG.

We chose to use the Mean Curvature Skeleton (MCS) on each of the protein environments because MCS is able to find more regions in a complex concave environment like these proteins. Thus, it will be better in assessing the effectiveness of the biasing methods in these particular environments.

The data for the protein environments were obtained from the Protein Data Bank (PDB) [5], and its corresponding geometric model was extracted using UCSF Chimera [6]. The ligand was obtained in a similar manner, extracting its geometric model from a PDB file with our own custom script.

As for the experiment setup, we ran all three methods for four different proteins (3fbw, 4fwb, 3rk4, hzg), which all bind to the same type of ligand (3KP). For each protein we ran all three strategies for 10 different seeds. The planner will stop building the roadmap once a specified number of nodes are built outside the hull of the protein. The resulting roadmap and tunnels that were explored will then be observed and analyzed.

Result The 10 runs done for each strategy and their corresponding protein were analyzed together. Table 2 displays the results for the planning step for each of the runs. The median of the runtime, the size of the roadmap in nodes and edges, and the number of tunnels found were calculated across each of the 10 seeds. The bolded values in the table are the energy biasing values to help highlight the differences between the strategies. We also observed which tunnels the strategies explored and how their characteristics compare to each other. Figure 6 displays the roadmaps built from three example runs from the three strategies. The energy and clearance biasing strategy are displayed with the annotated skeleton to see the biasing clearer.

Protein	Strategy	Runtime	Nodes	Edges	Tunnels
3fbw	default	173	644	2878	12
	energy	**121**	**413**	**2534**	**12**
	clearance	160	616	2546	10
3rk4	default	273	772	3202	12
	energy	**318**	**693**	**2683**	**11**
	clearance	329	428	1904	9
4fwb	default	125	559	2421	8
	energy	**125**	**492**	**2183**	**9**
	clearance	122	472	2389	12
4hzg	default	123	455	1907	8
	energy	**169**	**350**	**1649**	**11**
	clearance	141	454	1984	11

Table 2: Experiment Results - Comparison of strategies

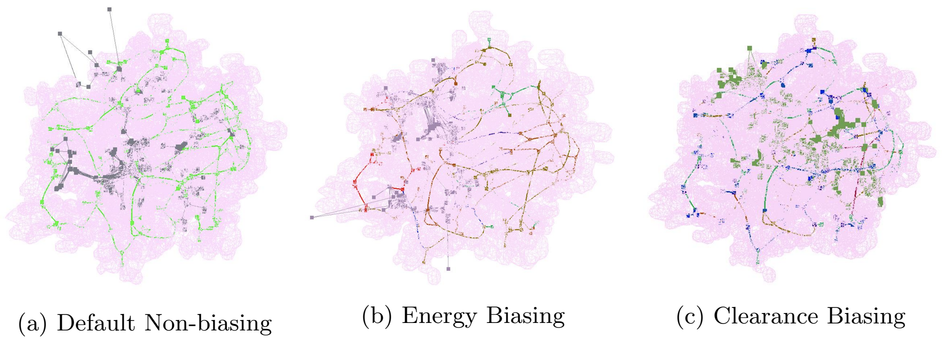

(a) Default Non-biasing (b) Energy Biasing (c) Clearance Biasing

Fig. 6: Example roadmaps built from default, energy biasing and clearance biasing planning strategies

Discussion As shown in table 2, each of the strategies finished in similar times for each environment. In that time, the bias strategies created a smaller, tighter roadmap. As shown in figure 6, both the clearance and energy bias strategies have explored towards favorable regions of the skeleton, which are colored in warmer red or orange colors. The biased roadmaps congregate around these low energy, high volume areas, indicating that the strategies were able to find more of these favorable tunnels.

The number of tunnels found throughout the 10 seed runs are relatively around the same across all strategies, and the runtime of all three are also around the same. Since we gave all three strategies the same threshold they must finish by, this indicates that the biasing strategies are actually more efficient in finding these specified tunnels. The tunnels that the bias strategies found were more meaningful in the sense that the tunnels was higher ranked in the given bias metric. This indicates that our biasing strategies are able to find more meaningful tunnels in less time.

5 Conclusions

We introduced using metrics such as clearance and energy to improve the exploration process of motion planning problems. Our main contribution is the use of workspace properties available to skeleton-guided planners to improve the planning process. The Bias-guided method is dependent on the workspace skeleton, and it performs best on Motion Planning problems which have strong relations to its workspace topology.

The Bias-guided method allows us to find the most desirable paths faster and to target exploration to narrow passages using clearance value in robotics and energy value in protein environments. The experiments above demonstrate how our method works in both environments while also highlighting its features and bottlenecks.

In theory, more workspace related metrics can be designed for any motion planning problem that would benefit from exploiting such metric during planning.

5.1 Future Work

We plan to extend the clearance metric to other motion planning applications like animation and Image-guided Medical Needle Steering. We also plan to test the clearance-bias method in real-world navigation experiments.

In regards to the protein environments, we are working on tuning our biasing strategy to be able to rank the protein tunnels on how easily they can be discovered as the planner keeps exploring.

Acknowledgement

We want to express our gratitude to Nancy M. Amato (faculty mentor), Shawna Thomas (research specialist advisor), and Diane Uwacu (graduate student mentor), for their support and guidance throughout this project.

This research is supported in part by the Computing Research Association-Women (CRA-W) Distributed Research Experience for Undergraduate (DREU) and by the Computer Science Department at the University of Illinois at Urbana-Champaign.

References

1. Latombe, J.C.: Motion planning: A journey of robots, molecules, digital actors, and other artifacts. Int. Journal of Robotics Research 18(11), 1119–1128 (1999)
2. Kavraki, L.E., vestka, P., Latombe, J.C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Automat. 12(4), 566-580(August 1996)
3. Kaushik, S., Marques, S.M., Khirsriya, P., Paruch, K., Libichova, L., Brezovsky J., Prokop, Z., Chaloupkova, R., Damborsky, J.: Impact of the access tunnel engineering on catalysis is strictly ligand specific. Federation of European Biochemical Societies Journal 285(8), 1456-1476(2018)
4. Denny, J., Sandstrom, R., Bregger, A., Amato, N.M.: Dynamic region-biased exploring random trees. In: Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR). San Francisco, CA (December 2016)
5. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T., Weissig, H., Shindyalov, I., Bourne, P.: The protein data bank. Nucleic Acids Research 28(1) 235-242 (2000)
6. Pettersen, E.F., Goddard, T.D., Huang, C.C., Counch, G.S., Greenblatt, D.M., Meng, E.C., Ferrin, T.E.: Ucsf chimera: A visualization system for exploratory research and analysis. Journal of Computational Chemistry 25(13), 1605-16012 (2004)
7. Tagliasacchi, A., Alhashim, I., Olson, M., Zhang, H.: Mean curvature skeletons. Eurographics Symposium on Geometry Processing 2012 27(1) (2012)

8. Kala, R.: Rapidly exploring random graphs: motion planning of multiple mobile robots. Advanced Robotics 27(14), 1113-1122(2013), https://doi.org/10.1080/01691864.2013.805472
9. Levitt, M.: Protein folding by restrained energy minimization and molecular dynamics. J. Mol. Biol. 170, 723-764 (1983)