Dynamic Sampling for N-Link Chain Using Workspace Segmentation

Dhrumil Sandeep Kotadia
dkotadia@wpi.edu

Dhiraj Kumar Rouniyar
dkrouniyar@wpi.edu

***Abstract*—Diverse sampling techniques are commonly employed in probabilistic planning approaches to determine paths. However, no single sampling method universally excels in all scenarios. Challenges arise when existing sampling methods underperform across an entire map. In response, we propose a novel solution by partitioning the workspace and employing distinct sampling methods in different map segments. Leveraging machine learning, our approach identifies the most effective sampling method for each map segment, resulting in a non-uniform dynamic sampling strategy. This innovative method eliminates the need for oversampling across the entire map, offering an efficient path-finding solution. The project involved an N-Link chain as the robotic model in our project.**

I. Introduction

In the realm of probabilistic planning approaches aimed at path determination, a diverse array of sampling methods is frequently utilized. This multifaceted landscape acknowledges the inherent challenge that no single sampling method universally excels across all conceivable scenarios. The dynamic interplay between varied environmental settings and diverse robotic platforms underscores the necessity of tailoring the selection of sampling methods to specific contexts for optimal outcomes.

Complications emerge when confronted with workspaces that encompass a synthesis of multiple disparate scenarios. Such workspaces pose a unique set of challenges, rendering conventional, uniform sampling methods less effective in achieving desired results. In such complex scenarios, the inadequacy of existing uniform sampling approaches becomes evident, prompting the need for extensive oversampling throughout the entirety of the workspace to compensate for their limitations. This, in turn, leads to inefficiencies and resource-intensive processes that could otherwise be mitigated through more nuanced and adaptive sampling strategies.

The subsequent sections of this report delve into a novel solution that addresses these challenges by advocating for a segmented approach to workspace analysis. By strategically deploying different sampling methods in various segments of the workspace, we aim to optimize the outcomes in a context-dependent manner. The ensuing discussion outlines the incorporation of machine learning techniques to discern and prescribe the most effective sampling methods for each distinct segment of the workspace. This dynamic and context-aware sampling strategy seeks to alleviate the need for indiscriminate oversampling across the entire workspace, thereby enhancing the efficiency and precision of probabilistic planning for robotic pathfinding.

II. Related Works

As we delve into the intricacies of probabilistic planning and pathfinding, it becomes important to explore innovative approaches that rise above the limitations of uniform sampling. There have been numerous attempts for development of non uniform sampling methods as they usually provide a much better output compared to uniform sampling. Many methods have been used for the same which include Workspace importance sampling [3], planning for similar obstacles [4] and biased sampling based on learning [1] among others. Other approaches have been explored in the past that focus on adaptive workspace bias sampling and segmenting the workspace [5] but do not utilize the learning approach to optimize the efficiency.

III. Proposed Method

The procedural details of the proposed method are illustrated in Figure 1. To train the machine learning model effectively, the initial step involves the creation of a comprehensive dataset. This dataset is meticulously generated through the careful selection of diverse maps and sampling methods as discussed below.

A. Maps

A total of 23 unique 2D workspace maps have been created with different number of obstacles in different locations. Some of them can be sen in figure 2

One of the most important considerations while generating the maps was the number of obstacles present in each map. The maps are generated in such a way that they can be classified based on the number of obstacles. This classification helps in identifying the the sampling strategy and the number of samples for the segments of the test map.

Given the asymmetry of the maps, the positioning of the kinematic chain's base emerges as a crucial factor. The presence or absence of formidable obstacles along the kinematic chain's trajectory depends significantly on the specific characteristics of each map. To address this variability and obtain a comprehensive overview of the map's challenges, the base of the kinematic chain is strategically placed in eight distinct locations across the map. Each location is associated with unique start and goal configurations, contributing to a

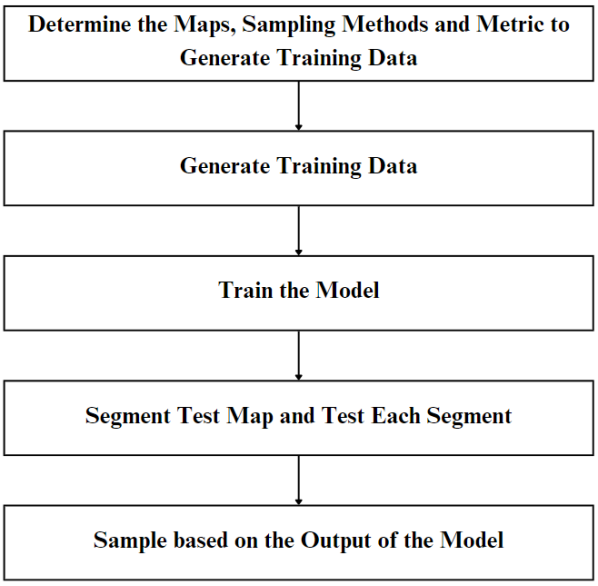

Fig. 1. The Process

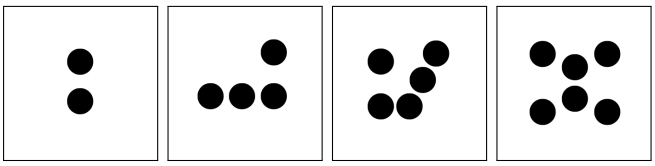

Fig. 2. Sample Maps created to generate training data

more nuanced understanding of the sampling outcomes across different regions of the map. This approach ensures a more robust and generalized assessment of the sampling strategy's efficacy by accounting for variations in obstacle density and spatial arrangement within the diverse map scenarios.

B. Sampling Methods

A total of 4 sampling methods have to be selected. PRM [2] has been selected as the planner. To generate the training data, all 4 sampling methods have been executed in all of the scenarios. The following sampling methods have been utilised for this purpose:

- Uniform Random Sampling
- Gaussian Sampling
- Bridge Sampling
- Custom Chain Sampling

Custom Chain Sampling: The Custom Chain Sampling Strategy employed in this project operates on the principle of generating random uniform samples and utilizing them unless a collision is encountered. In the event of a collision with a generated sample, the strategy initiates a two-fold response. Firstly, it generates an additional set of five samples by introducing variations in the angles of the base joint, with each sample deviating by 5 degrees. This is evident in figure 3.

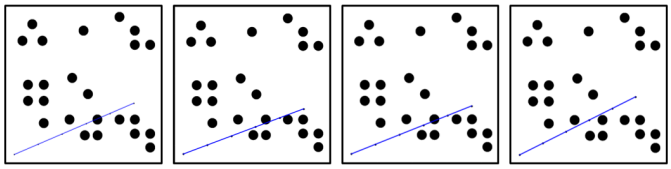

Fig. 3. Example of modifying the angle of the base joint of the Kinematic Chain

Secondly, the chain is retracted back towards the base and five points are sampled along that trajectory. Figure 4 visually encapsulates this process. From the pool of ten samples generated, only those that successfully avoid collisions are ultimately utilized in the sampling strategy.

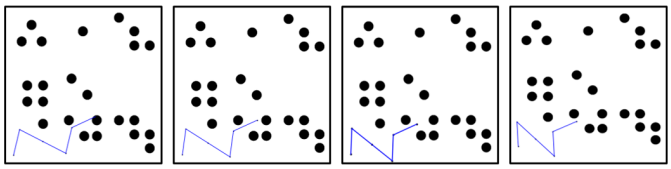

Fig. 4. Example of Retracting the Kinematic Chain back towards the base

The essence of the sampling strategy revolves around two fundamental concepts. Firstly, in regions with high obstacle density where the chain faces collision, a slight adjustment to the base angle can potentially result in a configuration that avoids collision. Secondly, when confronted with an obstacle, the chain is incapable of traversing it directly; instead, it needs to retract back towards the base and circumvent the obstacle. Consequently, having an increased number of samples distributed between the base and obstacles proves advantageous for navigating the workspace effectively, facilitating the strategy's adaptability to intricate environments.

During the generation of training data, the sampling methods remain unaltered; however, modifications are introduced while implementing on the test map to incorporate projections, thereby reducing complexity and sampling time. In the test map scenario, a substantial set of 50,000 uniform random samples is initially extracted across the entire map without collision checking. Subsequently, the forward kinematics process is employed to ascertain the position of the end-effector corresponding to each sample. This end-effector position is then utilized to identify the specific segment within the overall map where the sample is situated. Depending on the designated sampling method and the requisite number of samples in that segment, the current sample undergoes further adjustments. In cases where the sampling method is uniform, a collision check is executed for the sample, and it is added if no collision is detected. For Gaussian sampling, collision checking is carried out, and if a collision occurs, a new sample with a Gaussian offset, free from collision, is calculated and employed. In bridge sampling, collision verification is performed, and if a collision is encountered, an alternative sample with a Gaussian offset is generated. If both samples demonstrate collision,

an interpolating sample devoid of collision is utilized. This hierarchical approach, utilizing end-effector positions at the top level to joint angles at the lowest level, eliminates the need for computing inverse kinematics for each sample.

C. Map Evaluation Metric

In the process of generating training data, all sampling strategies are systematically applied to each map within the project scope. To evaluate the effectiveness of each sampling strategy for a given map, a defined metric, specifically the number of nodes, is employed. The rationale behind this metric lies in its reflection of the minimum number of nodes required for a sampling strategy to successfully establish a path within a given map. The developed algorithm iteratively generates 10 samples for each sampling strategy, examining whether a path is discovered. If a path is not found, an additional 10 samples are generated, and this iterative process continues until a path is successfully identified. The corresponding count of samples needed to achieve a viable path is meticulously recorded for each sampling strategy and map pairing, providing valuable insights into the efficiency and adaptability of the respective strategies across different map scenarios.

The dataset creation process adheres to the procedural flow outlined in Figure 5. It begins with the selection of a map, followed by the systematic application of all sampling methods across various base locations. Once a viable path is identified, the corresponding data is stored in a text file. Subsequently, an average of the number of samples required for each map and each sampling method is computed from this data file. The maps are then categorized into distinct classes based on the number of obstacles present, such as 2-obstacles, 3-obstacles up to 6-obstacles. For each class, the average number of necessary nodes is determined for each sampling method. This information serves as the foundation for identifying the optimal number of samples and the most effective sampling method for each segment of the test map, offering valuable insights into the adaptability and efficiency of sampling strategies across different obstacle scenarios.

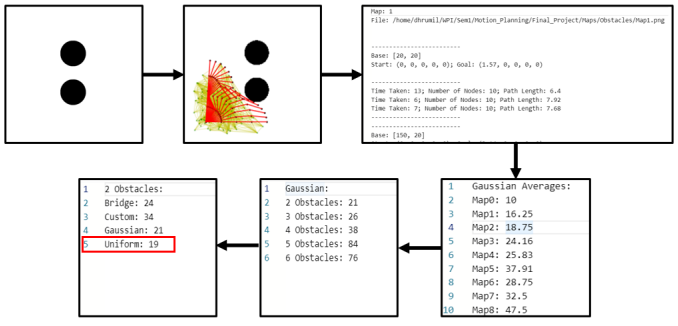

Fig. 5. Process of generating the dataset

IV. Learning

The planner needs to identify the map to apply a sampling method that suits it. Therefore, deep learning models designed for image classification tasks are utilized, each involving distinct procedures to adhere to.

A. Dataset Preprocessing

The dataset prepared through compution of viable path as explained in the Figure 4 is then customised to increase training samples and variations in the dataset. Prior to training the model with the dataset, it is necessary to undergo pre-processing using data augmentation to enhance the model's generalization. Consequently, the dataset is expanded using techniques such as rotations, flips, zooms, and shifts. Additionally, to introduce variations in the dataset, grayscale images featuring a white background and black-colored obstacles is inverted to their opposite colors and transformed into different colors. This diversification in the dataset through data augmentation techniques aids in enhancing the model's robustness. The ultimate dataset was effectively increased to 100 samples in total, subsequently segmented into distinct obstacle classes, and further partitioned into training and test sets for each class.

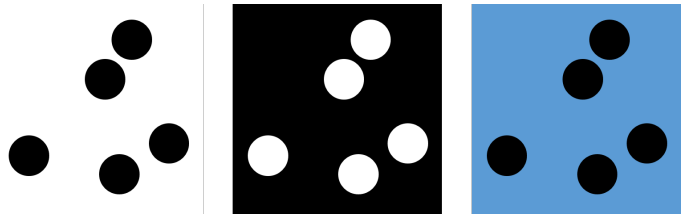

Fig. 6. Data augmentation with colours

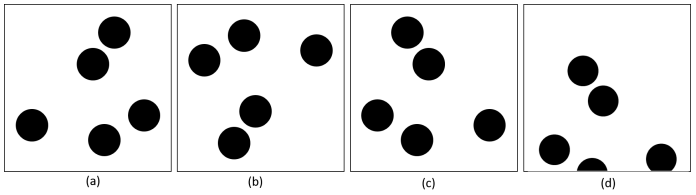

Fig. 7. Data augmentation with orientation, (a)Base image (b)Rotated (c)Flipped (d)Shifted

B. Deep Learning model selection

With an aim to classify images to identify the obstacle type and corresponding sampling strategy, four models namely Softmax Regression, Convolutional Neural Network (CNN), Residual Network (ResNet) and Capsule Networks (CapsNet) were selected considering their popularity in image classification tasks. At first, Softmax Regression model was selected to observe its accuracy on the prepared dataset with 12 obstacle scenarios. It was observed using image of shape 300,300,3 (h,w,RGB) resulted numpy array shape of 270000 one dimensional array. This could increase the model complexity if high number of training data set is used. To solve the issue, all the corresponding numpy array of the images in dataset prepared was reshaped to low dimension, however, it was observed the input dataset looses significant features as shown in Figure 8

and also the model accuracy was observed to be low as shown below in Figure 9.

The model accuracy could be increased through transfer

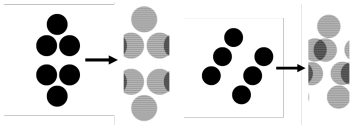

Fig. 8. Loss of features due to down reshaping

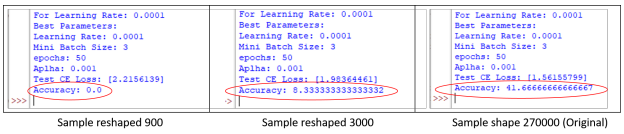

Fig. 9. Accuracy loss due to down reshaping

learning techniques and deep models and hence ResNet50 and CapsNet models were studied. These models are of complex architectures and have demonstrated superior performance on a wide range of challenging image classification tasks, however, these models have a larger number of parameters, making them more prone to overfitting, especially when the dataset is small as in our case and also the computational time is high. Therefore, to address our particular task of multi-class image classification with a limited dataset, we iteratively crafted a simpler model using a Convolutional Neural Network (CNN) with custom layers and parameters.

The custom CNN architecture consists of three convolutional layers with max-pooling followed by a flatten layer and two dense layers for classification. The convolutional layers utilize (3, 3) filters, and max-pooling layers have (2, 2) pool sizes. ReLU activation is applied to convolutional layers, and softmax is used for the output layer. This design strikes a balance between model complexity and interpretability. With a batch size of 1 and image preprocessing involving rescaling to [0, 1], the architecture aims for efficient memory usage. Although lacking explicit regularization techniques and transfer learning, the model uses categorical crossentropy loss, softmax activation for the output layer, and Adam optimizer. Following the initial development of the model, preliminary evaluations were conducted using a smaller dataset for sanity checks. Subsequently, the model was retrained with an expanded dataset comprising 100 samples. The resulting model demonstrated an accuracy of approximately 89%, successfully predicting obstacle classes, as illustrated in Figure 10.

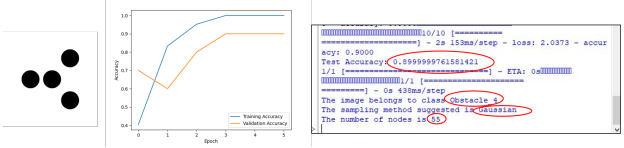

Fig. 10. Model accuracy and predicted Obstacle class, $Obstacle\_4$

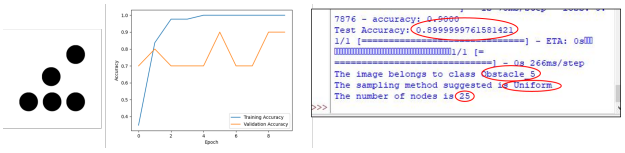

Fig. 11. Model accuracy and predicted Obstacle class, $Obstacle\_5$

V. Results and Evaluation

To test the validity of the mentioned approach, the test map is taken as shown in figure 12.

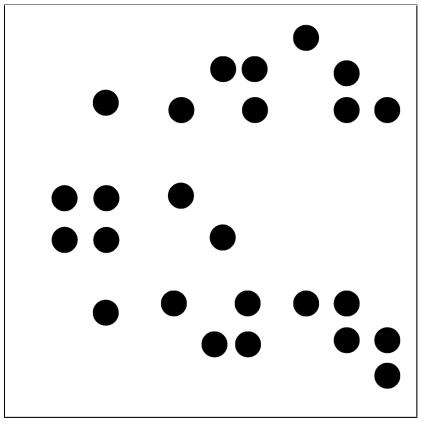

Fig. 12. The test map created for final Implementation

A 5 link chain with Link lengths 150 is considered as the robot and is placed at the bottom left corner of the map. The test map is divided into 9 equal square parts and each part is given to the model as testing data. Based on the training data provided to the model, the model provided the number of nodes and sampling methods for each section of the map. The resulting sampling is shown in figure 13.

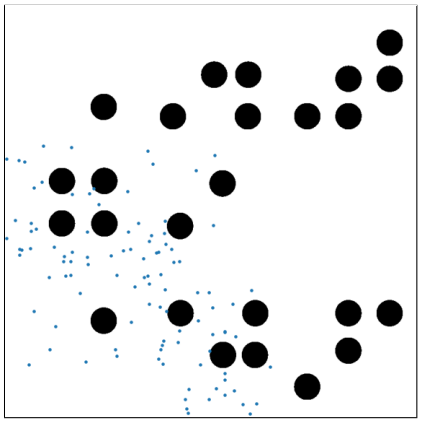

Fig. 13. Result of Dynamic sampling on test map

While the initial impression may be that the sampling process is arbitrary, a closer examination in Figure 14 reveals a structured division of samples based on model-informed data. Notably, in the segment featuring four obstacles, the model recommends Gaussian sampling with 38 samples. In segments

with one and two obstacles, the model advocates for Uniform sampling with 10 and 19 samples, respectively. The labeled segments in Figure 14 are denoted as U10, G38, and U19, representing Uniform sampling with 10 samples, Gaussian sampling with 38 samples, and Uniform sampling with 19 samples, respectively. This discernible pattern underscores the strategic integration of sampling methods tailored to specific obstacle scenarios as guided by the model-derived insights.

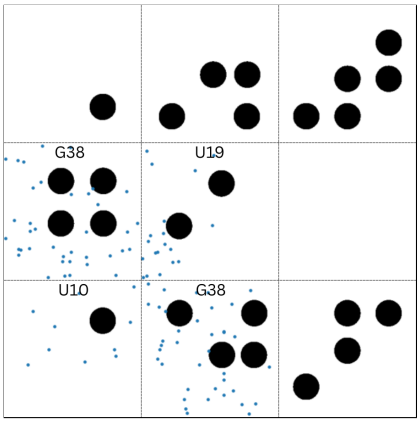

Fig. 14. Sampling in each segment

Figures 13 and 14 further illustrates the impact of dynamic sampling, revealing a strategic allocation of samples that places a greater emphasis on areas posing potential traversal challenges. Notably, regions perceived as more challenging to navigate receive a higher concentration of samples, while those anticipated to be more straightforward have a comparatively lower sample density. The exact number of samples per segment provided by the proposed method may not provide the path. In that case, increasing the number of samples while keeping the ratio of the number of samples between the segments constant provides a better probability of finding a path. Figure 15 shows the boxplot of the number of nodes required to find a path in the given map.

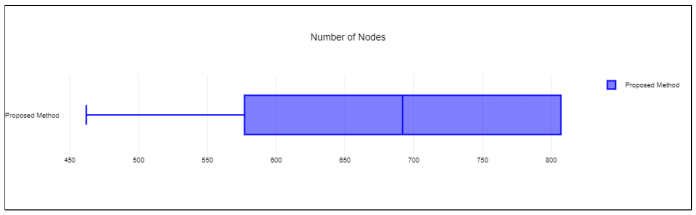

Fig. 15. Number of Nodes required to find a path using the proposed method

A more insightful evaluation of the proposed method's performance in contrast to existing methods can be derived by comparing the number of nodes required to establish a path through uniform sampling. Although obtaining a path through uniform sampling is a time-consuming process, hindering direct comparison, it is noteworthy that even before successfully identifying a path, the number of nodes has already surpassed 1000. This early accumulation of nodes provides a preliminary indication of its potential effectiveness in navigating complex environments.

VI. Conclusion

In conclusion, this study has presented a comprehensive exploration of a dynamic sampling strategy for probabilistic planning in robotic pathfinding. By strategically adapting sampling methods based on the specific challenges encountered within different segments of the workspace, our proposed approach effectively addresses the issue of indiscriminate oversampling across the entire workspace, as it ensures that an ample number of samples are concentrated in areas presenting navigation difficulties. This nuanced strategy optimizes the allocation of resources, directing sampling efforts where they are most needed for effective pathfinding in intricate or challenging regions.

VII. Limitations and Future Work

The existing implementation exhibits several limitations and offers ample opportunities for enhancement. A primary area of improvement pertains to the dataset generation process. Enhancing the generation methodology is crucial for refining the training of the learning model, ultimately yielding superior results. Additionally, the current procedural flow is somewhat disjointed, requiring manual entry of the learning model's output for testing purposes. A more streamlined interface can be achieved by creating a wrapper, thus facilitating a seamless integration of the entire process.

Furthermore, the variety of sampling methods employed presents an additional avenue for improvement. Introducing and exploring additional sampling methods holds the potential to enhance overall efficiency and effectiveness in robotic pathfinding. Equally important is the optimization of dataset preparation, constituting a critical aspect of the project that warrants refinement.

In summary, the key areas for substantial improvement revolve around the meticulous generation of relevant and reliable training data and the exploration and integration of diverse sampling methods. By addressing these aspects, the project can advance significantly, resulting in a more robust and versatile probabilistic planning approach for robotic pathfinding.

VIII. Break-Down Contribution of Team Members

- Both
  - Deciding and finalizing the problem statement and the approach
- Dhrumil Kotadia
  - Deciding Sampling strategies, creating obstacle maps and deciding parameters for sampling data generation.
  - Creating custom chain sampling strategy
  - Generating data for training the machine learning model
  - Deciding and implementing the methodology to sample with different sampling methods in different segments of the map.

  - Generating final results.
- Dhiraj Kumar Rouniyar
  - Preparing custom dataset for model training through data preprocessing
  - Deciding data augmentation technique
  - Deciding model parameters like solver, loss function, etc with extensive literature review
  - Iteratively implementing different models and judging model for binary classification job
  - Implementing Softmax Regression and CNN
  - Deciding architecture depth and layers through iterations
  - Deciding convolution and max pooling filters for effective training of binary images

REFERENCES

[1] B. Ichter, J. Harrison, and M. Pavone. Learning sampling distributions for robot motion planning. 2018.

[2] L. Kavraki, S. P., J. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. 1996.

[3] H. Kurniavati and D. Hsu. Workspace importance sampling for probabilistic roadmap planning. 2004.

[4] J.-M. Lien and Y. Lu. Planning motion in environments with similar obstacles. 2009.

[5] M. Zucker, J. Kuffner, and J. A. Bagnell. Adaptive workspace biasing for sampling-based planners. 2008.