The Thirty-First International Florida
Artificial Intelligence Research Society Conference (FLAIRS-31)

Efficient Real-Time Robot Navigation Using Incremental State Discovery Via Clustering

Olimpiya Saha, Prithviraj Dasgupta
Computer Science Department
University of Nebraska, Omaha
USA *

Abstract

We consider the problem of robot path planning in an initially unknown environment where the robot does not have access to an *a priori* map of its environment but is aware of some common obstacle patterns along with the paths that enable it to circumnavigate around these obstacles. In order to autonomously improve its navigation performance, the robot should be able to identify significant obstacle patterns and learn corresponding obstacle avoidance maneuvers as it navigates through different environments in order to solve its tasks. To achieve this objective, we propose a novel online algorithm called *Incremental State Discovery Via Clustering (ISDC)* which enables a robot to dynamically determine important obstacle patterns in its environments and their best representations as combinations of initially available basic obstacle patterns. Our results show that ISDC when combined with our previously proposed navigation technique was able to identify significant obstacle patterns in different environments in a time effective manner which accelerated the overall path planning and navigation times for the robots.

Introduction

A crucial aspect in real-time robot path planning is to enable robots to determine obstacle boundaries so that they can avoid collisions. Many existing path planning techniques (Arslan 2016; Arslan, Guralnik, and Koditschek 2016; Ott and Ramos 2013; Ravankar et al. 2012) use supervised learning techniques where robots are trained to recognize obstacle boundaries as features. However, supervised learning techniques require considerable human effort to label features and their properties. To identify features rapidly, in real-time and without human assistance, it would be useful if a robot could autonomously learn features from its perceived sensor data during navigation. A possible solution to this problem is afforded by a clustering-based unsupervised learning technique called affinity propagation (Frey and Dueck 2007) that iteratively exchanges real-valued messages between data points to automatically cluster them. Although affinity propagation has been used extensively in different domains including biology, sensor networks, decision making in business and operational research, there has been limited utilization of clustering techniques in real-time robotics, possibly due to large time requirements of clustering algorithms that are not suitable for real-time operation of robots. In this paper, we attempt to speed up the clustering technique by proposing a framework called *Incremental State Discovery via Clustering (ISDC)* that combines option augmented meta-point affinity propagation (OMP-AP), an advanced variant of affinity propagation adapted for robot navigation, with a Fast Approximate K-Nearest Neighbor (K-NN) classifier (Muja and Lowe 2009). Our experimental results with Coroware Corobot robots within the Webots simulator for different robot navigation tasks in different environments, with different obstacles, validate that clustering obstacle features using ISDC reduces the overall planning time by 81%, total time by 64%, total distance by 8% and only reduces the average expected q-values by 27% in comparison to a previously proposed hierarchical navigation technique SMDPU-T (Saha and Dasgupta 2017). In order to explicitly assess the advantage of clustering in ISDC, we have analyzed its performance after the clustering process and found that ISDC on average reduces the planning time by 84%, total time by 74%, total distance by 25% and only decreases the average expected q-value achieved by 13% compared to SMDPU-T. In spite of the average reduction in the q-values, ISDC was able to achieve a final q-value which was 21% higher compared to the final q-value achieved by SMDPU-T.

Related Work

Clustering has been applied in robotics for designing scalable algorithms for controlling the motion of large groups of robots as motion coordination of large multi-robot teams is computationally challenging. Some of the major works in this area include (Ayanian, Kumar, and Koditschek 2011; Chaimowicz and Kumar 2007; Ogren 2004). In(Arslan 2016; Arslan, Guralnik, and Koditschek 2016), the authors have proposed a computationally efficient coordinated multi-robot motion design utilizing hierarchical clustering techniques. Works which specifically involve clustering with obstacle avoidance and robot path planning include (Imeson and Smith 2017; Mas and Kitts 2012). However, most of these methods utilize clustering techniques

*O. Saha is a PhD student and P. Dasgupta is a Professor in the Computer Science Department, University of Nebraska, Omaha (osaha,pdasgupta)@unomaha.edu

as offline computations in multi-robot paradigm. In contrast, in our work, we propose to utilize a fast clustering technique called meta-point affinity propagation (Frey and Dueck 2007; Ott and Ramos 2013) that will enable robots to autonomously learn appropriate representations of their environment dynamically by performing clustering in real-time. In (Ravankar et al. 2012), the authors propose a map-building technique for mobile robots using k-means clustering on laser sensor data. However, this technique suffers from the limitation of specifying the number of clusters *a priori* and is extremely sensitive to noise. Our work extends the line of research proposed in (Frey and Dueck 2007; Ott and Ramos 2013) but focuses explicitly on the incorporation of this fast clustering technique to augment the real-time navigation performance in robots by enabling them to determine the intrinsic environmental features across environments.

Problem Formulation

We consider the problem of navigation where a wheeled robot has to navigate efficiently in an initially unknown bounded environment composed of multiple complicated obstacles $\mathcal{Q}_{obs}$ and free space $\mathcal{Q}_{free}$. We consider that the robot does not possess any *a priori* map of the environment. Thus, the robot is unaware of the location and geometry of the obstacles. The primitive actions of the robot are represented in the continuous space as $a = (\psi, d)$ where ψ defines the desired bearing and d is the distance that the robot moves. Robots are also equipped with depth sensors (e.g. laser) which enable them to record point clouds when placed in close proximity of the obstacles. Robots are also assumed to be localized with respect to the environment using a localization device. The navigation task requires a robot to determine a collision-free path between the start and goal $(q_{start}, q_{goal}) \in \mathcal{Q}_{free}$. For this, the robot has to determine a sequence of actions that results in a collision free path when it encounters an obstacle in its path. ISDC utilizes a previously proposed hierarchical navigation technique SMDPU-T (Saha and Dasgupta 2017) as its underlying motion planner which combines semi-Markov decision process (SMDP)(Sutton, Precup, and Singh 1999) and Markov decision process with unawareness (MDPU)(Halpern, Rong, and Saxena 2010).

Option Augmented Meta-point Affinity Propagation

Affinity propagation (AP) (Frey and Dueck 2007) algorithm requires the similarity values between data points as its input. The similarity between data points is defined depending on the problem domain. To apply AP to our problem of real-time obstacle recognition, we need to capture features that are both scale and rotation invariant and are capable of appropriately capturing the overall shape of the laser scans. We investigate the applicability of *Viewpoint Feature Histogram (VFH)* (Rusu et al. 2010) as distinctive features for similarity calculation for clustering obstacle patterns. Rusu et al. achieved promising results in household object recognition using VFH as classification features and Fast Approximate K-NN as classifier. However, we believe clustering is an important step for the problem of robot navigation as unlike classification where the number of classes is known beforehand, in our problem, there can be previously unseen obstacle(s) encountered by the robot while solving the current navigation task. Under such a condition, the robot should be able to dynamically update the number of classes as and when required. VFH is an advanced variant of Point Feature Histogram (PFH) (Rusu et al. 2008) which evaluates the relative pan, tilt and yaw angles between the surface normals of the centroid and every point on an object and bins that results in a histogram. In order to compute the similarity between the histograms of two obstacle patterns which is needed by AP to create the clusters, we use histogram intersection kernel following $d(H_1, H_2) = \Sigma_{i=1}^{g} min(H_1^i, H_2^i)$ where g is the number of histogram bins and H_1, H_2 are the two histograms to be compared and H_1^i and H_2^i are respectively the values of H_1 and H_2 at the ith bin (Barla, Odone, and Verri 2003). From the input similarity information, a graph is constructed where the nodes represent the individual obstacle pattern and edges represent the similarity between pairs of obstacle patterns. Affinity propagation proceeds by interchanging two types of messages between connected nodes in the graph called *responsibility* and *availablity* (Eqns 1, 2).

$$r(i,k) = s(i,k) - \max_{k' s.t. k' \neq k} (a(i,k') + s(i,k')) \quad (1)$$

$$a(i,k) = \min(0, r(k,k) + \Sigma_{i' s.t. i' \notin \{i,k\}} \max(0, r(i',k))) \quad (2)$$

where $s(i,k)$ is the similarity score between obstacle patterns i and k. The exemplar of a data point is another data point which represents the previous data point appropriately and has the highest similarity to it in the feature space. Meta-point affinity propagation (MP-AP) (Ott and Ramos 2013) extends the application of affinity propagation to real-time environments where high-speed clustering needs to be performed. In MP-AP, obstacle patterns which are close in the feature space are grouped together and replaced by a single meta-point. As our objective is to improve the robot's navigation by determining suitable obstacle pattern through clustering, we define option augmented meta-points (OMP) which are similar to meta-points with added corresponding option taken by the robot while navigating around each obstacle pattern (each data point of the meta-point), resulting in option augmented MP-AP (OMP-AP). An option is a sequence of time-extended actions with an initial state $\mathcal{I}$, a policy π and a termination condition β guided by a decision making framework called semi-markov decision process (SMDP) (Sutton, Precup, and Singh 1999). MP-AP considers two types of meta-points called cluster-points m_i^c and noise-points m_i^n. The former denotes obstacle patterns that are used for the standard affinity propagation and noise-points are obstacle patterns which are ignored while the execution of the actual clustering happens. We consider the set of noise points as $N = \{m_i^n\}$ and the set of cluster points as $U = \{m_i^c\}$. A noise-point is considered cluster-point if it represents sufficient number of obstacle patterns. When one or more noise-points qualify as cluster-points, they are eliminated from N and included in U. Mathematically, this can

be represented using equations 3, 4 and 5.

$$N = N \setminus m_i^n \text{ s.t. } |m_i^n| \geq \theta_{min} \quad (3)$$

$$m_i^c = m_i^n \quad (4)$$

$$U = U \cup m_i^c \quad (5)$$

where θ_{min} denotes the minimum number of obstacle patterns required to become a cluster-point m_i^c from a noise-point m_i^n. ISDC consists of an initial training phase and a testing phase which are described next.

Training Phase for Creating Initial Obstacle Patterns and Options in SMDPs

In this phase, we allow the robots to perform path planning while navigating in simple environments with representative obstacles and capture the basic obstacle patterns. When a robot encounters an obstacle pattern p, it records the contour of it as a set of 2D way points $p = \{(x, y)\}$. We apply a variant of Meta-point affinity propagation (Ott and Ramos 2013) on the obstacle data accumulated during the training process to identify the initial set of clusters C_0 formed of similar obstacle patterns. Each data point in the cluster corresponds to an obstacle pattern. The number of clusters in C_0 corresponds to the set of initial obstacle patterns in the explored environments. The exemplar pattern p_e of each cluster in C denotes the obstacle pattern which best represents the entire cluster c. Each exemplar pattern p_e is added as a new state s to the state space of the MDPU $S_{M^u} = S_{M^u} \bigcup s$. Now we allow the robot to plan paths around each of these exemplar patterns p_e by varying the goal locations around them. The navigation around each exemplar obstacle pattern p_e is recorded as a set of 2D way points $\{(x, y)\}$ which form the states of an SMDP and each obstacle avoidance maneuver learned during the training phase creates an option for the SMDP for p_e. The locations of the robot $\{(x, y)\}$ where it first perceived the obstacle pattern form the set of initiation states $\mathcal{I}$ for the SMDP. The policy π for the SMDP gives the sequence of actions that the robot needs to follow at each state or location for successfully avoiding the perceived obstacle pattern p_e. Hence, each exemplar obstacle pattern p_e, when perceived, maps to the initial state (x, y) of the option o i.e. $\mathcal{F} : p \rightarrow (x, y)$ such that $(x, y) \in \mathcal{I}$. During this training phase, the robot thus records a set of options $\mathcal{O}$ along with the set of MDPU states S_{M^u} (exemplar obstacle patterns) which will be used by ISDC for subsequent state discovery and later on by SMDPU-T for planning robot navigation in the future. For our proposed framework, we also consider a unit pattern $free$ corresponding to the case where the robot does not perceive any obstacle and is potentially in an obstacle free zone in the environment. In contrast to the training phase of conventional reinforcement learning where the robot is trained to learn the best action at each state, in the training phase of our approach, the robot focuses on a sub problem which involves learning the best actions to circumnavigate an obstacle pattern in its environment.

Clustering Combined with Classification for Fast Obstacle Recognition

While the robot navigates in its environment, it keeps track of the recorded obstacle patterns at each step along with the identified exemplar pattern p_e. Finally, when the robot arrives at the *free* state by avoiding the obstacle, it combines all the intermediary obstacle patterns captured as cp. The robot also keeps track of the intermediary options that it executes and combines these options to form the initial option $co = \{o_i\}$. To guarantee real-time performance, we have to ensure that the robot is able to quickly recognize the encountered obstacle pattern as either one of the exemplar obstacle patterns or an unknown pattern. The K-NN classifier classifies the combined obstacle pattern cp into one of these classes. In the former case, the cluster c corresponding to the exemplar p_e is updated with the new obstacle pattern p. In the latter case, the data from the obstacle pattern is passed to the clustering system as a new observation. For our problem, each data point is actually a collection of 2D points representing the perceived obstacle contour. The new observation is identified by the clustering system into one of the following cases- (i) There is a suitable cluster-point present corresponding to the new observation, (ii) There is a suitable noise-point present for the given observation but no suitable cluster-point, (iii) No suitable cluster-point or noise-point can appropriately represent the new observation. For the first two cases, the new observation updates the corresponding cluster-point and noise-point respectively along with their associated statistics. The cluster-points are also stored in an outlier reservoir Z. The actual clustering is performed by executing standard affinity propagation (AP) algorithm on the cluster-points which is computation-intensive. Hence, to ensure real-time performance, AP is executed only at certain intervals. Following (Ott and Ramos 2013), we consider the number of cluster-points in the outlier reservoir Z as a condition for determining when to perform AP. If the number of meta-points in Z exceeds a certain threshold, AP is called to execute clustering. Every time AP is executed, there is a possibility that a new cluster $c_{new}$is created expanding the set of initial clusters $C = C \bigcup c_{new}$. The new exemplar pattern p_e^{new} corresponds to a newly discovered state of the overlying MDPU and is added to its state space $S = S \bigcup s_{new}$. This creates a new SMDP in the context of the newly discovered MDPU state s_{new}. The combined option co recorded corresponding to s_{new} forms its initial option and s_{new} is added to the initiation state $\mathcal{I}$ for co. The clustering can also alter the exemplar(s) of the existing clusters in which case the MDPU state space S_{M^u} is updated with the updated exemplar pattern(s). A fast approximate K-NN classifier (Muja and Lowe 2009) is trained with the features extracted from the exemplars of the new cluster(s) which helps in the future obstacle recognition as the robot encounters other obstacles. Fast approximate K-NN utilizes multiple randomized kd-trees (Friedman, Bentley, and Finkel 1977), which is a form of balanced binary search tree, for efficiently storing the training data and finds approximate nearest neighbors in a time effective manner especially in high dimensional data using priority search (Silpa-Anan and Hartley 2008).

Autonomous Obstacle Pattern Expansion Using ISDC

In this subsection we explain in details how the robot autonomously determines important obstacle patterns and ex-

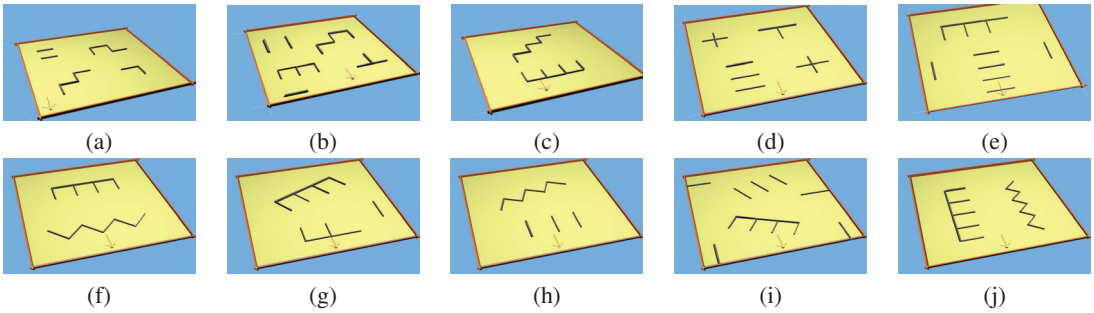


Figure 1: (a)-(j) Different environments used for testing proposed ISDC algorithm.

Algorithm 1: ISDC algorithm

Input: Perceived combined obstacle pattern cp, corresponding combined option co, set of cluster-points U, set of noise points N, Outlier reservoir Z

Output: Updated cluster set C, updated MDPU state space S_{M^u} and updated option set $\mathcal{O}$

1 $(U, N, Z, nn) \leftarrow$ OMP-AP(cp, U, N, Z)
2 **if** $Z.count > \zeta$ **then**
3 AFFINITY-PROPAGATION(Z)
4 $Z \leftarrow \varnothing$
5 $U \leftarrow \varnothing$
6 Update C
7 **repeat**
8 $s_{new} \leftarrow p_e^{new}$ s.t. p_e^{new} is the exemplar of c_{new}
9 $S_{M^u} \leftarrow S_{M^u} \bigcup s_{new}$
10 $O_{new} \leftarrow O_{new} \bigcup co$
11 $\mathcal{I} \leftarrow s_{new}$
12 **until** *all new cluster(s)* c_{new} *are covered*;
13 **repeat**
14 **if** *exemplar* $p_e^i \in c_i \neq s_i$ **then**
15 $s_i = p_e^i$
16 Update O for s_i with the associated option(s) co
17 **until** *all previously existing clusters* $c_i \notin C_0$ *are covered*;
18 Train K-NN with the features extracted from the exemplars of the updated clusters
19 Adjust policy π^u for each new state s_{new} and corresponding options O_{new}
20 Adjust policy π^u for the existing state(s) which got updated for executing AP

pands the MDPU state space S_{M^u} along with the associated option space $\mathcal{O}$ as it navigates in different environments. The pseudocode is given in Algorithm 1. When the robot ends up in a $free$ zone after avoiding the obstacles, then the OMP-AP algorithm is called with the combined obstacle pattern cp. Next the algorithm determines if AP needs to be performed by checking either if the outlier reservoir Z has exceeded its limit ζ (Line 2). If this condition is true, AP algorithm is called with the new data points as the set Z (Line 3). It returns the updated set of cluster-points U, noise-points N, outlier reservoir Z and the nearest neighbor set evaluated nn (Line 1). After AP is performed, the outlier reservoir Z and the set of cluster-points U are reset (Lines $4-6$) and the set of clusters C is updated. For each new cluster c_{new} created, the MDPU state space S_{M^u} is updated with the exemplar pattern p_e^{new} as s_{new} (Lines $8-9$). The associated set of options for the newly created MDPU state s_{new} is updated with the initial combined option co created while avoiding the detected obstacle. s_{new} is included in the set of initiation states $\mathcal{I}$ for option co in O_{new} (Lines $10-11$). Also for each cluster c_i previously existing in C which does not belong to the initial set of clusters C_0, it is checked if the exemplar p_e^i of each cluster c_i is same as the corresponding state s_i in the MDPU state space S_{M^u}. If these two are not the same, the MDPU state s_i is updated with the current exemplar as it is the best representation for the current cluster c_i (Lines $13-17$). After this, the K-NN classifier is trained with the features extracted from the exemplars of the updated clusters (Lines $18-19$). As the final steps, the algorithm adjusts the policy π^u with the newly added state s_{new} and options O_{new}. In case any change occurred to the exemplars of the existing clusters c_i outside the initially created set of clusters C_0, the ISDC also adjusts the policy π^u accordingly (Line 20).

Experimental Setup and Results

We have verified the performance of the ISDC algorithm using simulated corobot robot in the Webots simulator. The corobot is a four-wheeled ground robot equipped with a laser sensor with a 360° field of view and a view range of 2 m, a gps node and a compass. In order to assess the efficiency of our clustering-based ISDC algorithm, we have compared its performance with SMDPU-T (Saha and Dasgupta 2017) which learns each newly encountered obstacle patterns along with the corresponding options as the robot navigates in different environments. As SMDPU-T does not involve an explicit clustering process, we believe that its comparison with our algorithm will clearly illustrate the advantage of the clustering process with time.

For validating the comparative advantage of ISDC, we have used a set of 20 different simulated test cases where each

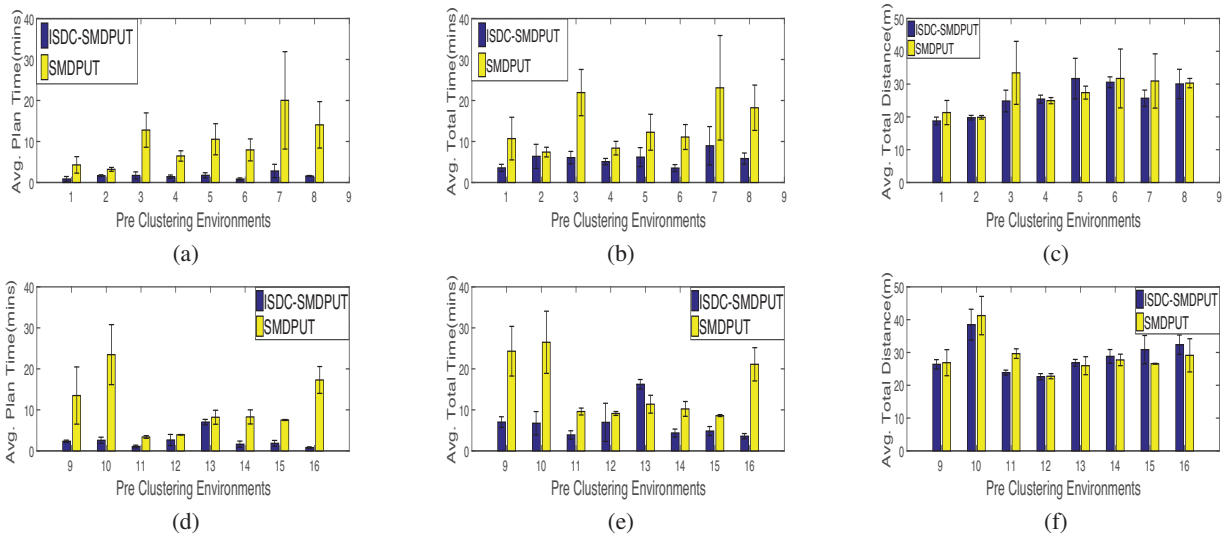


Figure 2: Planning Time, Total Time and Total Distance traveled by ISDC and SMDPU-T in (a)-(c) the first half of the pre clustering environments and (d)-(f) the second half of the pre clustering environments.

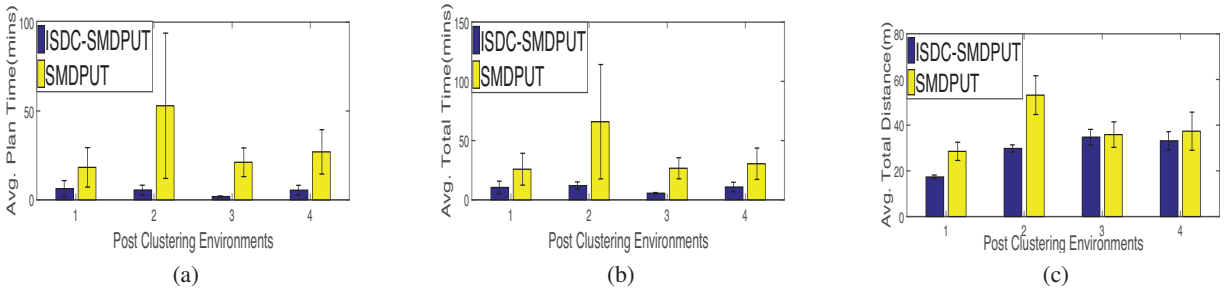
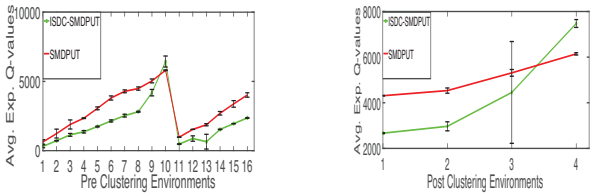


Figure 3: Planning Time, Total Time and Total Distance traveled by ISDC and SMDPU-T in the post clustering environments.

test case refers to a combination of distinct test environment (22×22 m^2) and start and goal locations. The robot was allowed to solve each test case using ISDC and SMDPU-T and the corresponding performance metrics were recorded. The first 16 of our test cases were used for experiments before ISDC performed OMP-AP clustering and the last 4 test cases were used for experiments after ISDC performed clustering and created new MDPU states. In order to account for randomness in both the algorithms, each test case was run 5 times making a total of 100 runs for each of the algorithms and the mean and standard deviation values were recorded. Figure 1 illustrates some of our test environments with representative obstacle patterns. We have compared the performance of ISDC and SMDPU-T in terms of both navigation as well as learning performance. For evaluating the navigation performance, we have reported three standard navigation metrics- planning time, total time and total distance. Planning time refers to the cumulative time that the robot expends in order to classify an encountered obstacle pattern into an MDPU state and select a suitable option from the available options for that state. Total time combines planning time and the time taken by the robot to perform navigation by following the recommended option. Total distance refers to the total distance covered by the robot to reach the goal from its start location. For assessing the comparative learning performance of ISDC, we have recorded the average expected Q-value achieved by the robot at the end of each test case. As ISDC utilizes a reinforcement learning-based motion planning framework SMDPU-T, we believe the average expected Q-value captures the learning performance of the algorithm in a succinct manner.

Figure 2 illustrates the comparative navigational performance of ISDC and SMDPU-T in the pre clustering environments (Test cases $1-16$). It can be observed from the figure that just with the contribution of Fast K-NN, ISDC performs better than SMDPU-T in terms of planning and total time and achieves comparable total distance for majority of the test cases. From our analysis of the navigation metrics in the pre clustering environments, we found that ISDC on average takes 18% of planning time, 36% of total time and 91% of total distance compared to SMDPU-T. Figure 3 illustrates the comparative navigational performance of ISDC and SMDPU-T in the post clustering environments (Test cases $17-20$). It is evident from the figure that with



(a) (b)

Figure 4: Average expected Q-values achieved by ISDC and SMDPU-T in (a)pre clustering environments and (b) post clustering environments.

the combined effects of high speed clustering and classification ISDC offers competitive advantage over SMDPU-T with respect to planning time, total time and total distance. From our analysis of the navigation metrics in these test cases, we found that ISDC takes 15% planning time, 26% total time and 74% total distance compared to SMDPU-T. From a comparative analysis of pre clustering and post clustering data, it can be observed that the combined effects of OMP-AP clustering and Fast Approximate K-NN on average leads to 3% reduction in planning time, 10% reduction in total time and 17% decrease in total distance covered. The comparative learning performance of ISDC and SMDPU-T is illustrated in Figure 4. It can be observed from the figures that although SMDPU-T on average achieves a higher average expected Q-value, however, the differences in Q-values between the two algorithms reduce almost consistently from pre clustering (Test cases $1-16$) to post clustering test cases (Test cases $17-20$). It can be observed that in the latter test cases this difference sharply reduces and ISDC eventually achieves an average expected Q-value which is 21% higher than SMDPU-T. From our analysis we have found that overall ISDC on average achieves 72% of the average expected Q-value and achieves 86% of the same exclusively in the post clustering test cases.

Conclusions

In this paper, we proposed a novel algorithm called *Incremental State Discovery Via Clustering (ISDC)* which integrates a fast, real-time clustering-based technique called option augmented Meta-point Affinity Propagation (OMP-AP) and a Fast Approximate K-Nearest Neighbor (K-NN) classifier. Our experimental results validate that the combined effects of high speed clustering and classification in *ISDC* reduces the overall path planning and navigation times in the robot along with reduced total distance and leads to eventual improvement in its learning performance. In future we plan to assess the performance of the ISDC algorithm on a hardware robot with higher amounts of pre and post clustering data.

References

Arslan, O.; Guralnik, D. P.; and Koditschek, D. E. 2016. Coordinated robot navigation via hierarchical clustering. *IEEE Transactions on Robotics* 32(2):352–371.

Arslan, O. 2016. *Clustering-Based Robot Navigation and Control*. PhD dissertation, University of Pennsylvania.

Ayanian, N.; Kumar, V.; and Koditschek, D. 2011. Synthesis of controllers to create, maintain, and reconfigure robot formations with communication constraints. In *Robotics Research*. Springer. 625–642.

Barla, A.; Odone, F.; and Verri, A. 2003. Histogram intersection kernel for image classification. In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, volume 3, III–513. IEEE.

Chaimowicz, L., and Kumar, V. 2007. Aerial shepherds: Coordination among uavs and swarms of robots. *Distributed Autonomous Robotic Systems 6* 243–252.

Frey, B. J., and Dueck, D. 2007. Clustering by passing messages between data points. *science* 315(5814):972–976.

Friedman, J. H.; Bentley, J. L.; and Finkel, R. A. 1977. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS)* 3(3):209–226.

Halpern, J. Y.; Rong, N.; and Saxena, A. 2010. Mdps with unawareness. *arXiv preprint arXiv:1006.2204*.

Imeson, F., and Smith, S. L. 2017. Clustering in discrete path planning for approximating minimum length paths. In *American Control Conference (ACC), 2017*, 2968–2973. IEEE.

Mas, I., and Kitts, C. 2012. Obstacle avoidance policies for cluster space control of nonholonomic multirobot systems. *IEEE/ASME Transactions on Mechatronics* 17(6):1068–1079.

Muja, M., and Lowe, D. G. 2009. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP 2009 - Proceedings of the Fourth International Conference on Computer Vision Theory and Applications, Lisboa, Portugal, February 5-8, 2009 - Volume 1*, 331–340.

Ogren, P. 2004. Split and join of vehicle formations doing obstacle avoidance. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 2, 1951–1955. IEEE.

Ott, L., and Ramos, F. 2013. Unsupervised online learning for long-term autonomy. *The International Journal of Robotics Research* 32(14):1724–1741.

Ravankar, A. A.; Hoshino, Y.; Emaru, T.; and Kobayashi, Y. 2012. Robot mapping using k-means clustering of laser range sensor data. *Bulletin of Networking, Computing, Systems, and Software* 1(1):pp–9.

Rusu, R. B.; Marton, Z. C.; Blodow, N.; and Beetz, M. 2008. Learning informative point classes for the acquisition of object model maps. In *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on*, 643–650. IEEE.

Rusu, R. B.; Bradski, G.; Thibaux, R.; and Hsu, J. 2010. Fast 3d recognition and pose using the viewpoint feature histogram. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2155–2162. IEEE.

Saha, O., and Dasgupta, P. 2017. Real-time robot path planning around complex obstacle patterns through learning and transferring options. In *Autonomous Robot Systems and Competitions (ICARSC), 2017 IEEE International Conference on*, 278–283. IEEE.

Silpa-Anan, C., and Hartley, R. 2008. Optimised kd-trees for fast image descriptor matching. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 1–8. IEEE.

Sutton, R. S.; Precup, D.; and Singh, S. 1999. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence* 112(1):181–211.