CoMBiNED: Multi-Constrained Model Based Planning for Navigation in Dynamic Environments

Harit Pandya [1] **Rudra P.K. Poudel** [1] **Stephan Liwicki** [1]

Abstract

Recent model based planning approaches have attained a huge success on Atari games. However, learning accurate models for complex robotics scenarios such as navigation directly from high dimensional sensory measurements requires a large amount of data and training. Furthermore, even a small change on robot configuration such as kinodynamics or sensor in the inference time requires re-training of the policy. In this paper, we address these issues in a principled fashion through a *multi-constraint model based online planning* (CoMBiNED) framework that does not require any retraining or modifications on the existing policy. We disentangle the given task into sub-tasks and learn dynamical models for them. Treating these dynamical models as soft-constraints, we employ stochastic optimisation to jointly optimize these sub-tasks on-the-fly at the inference time. We consider navigation as central application in this work and evaluate our approach on publicly available benchmark with complex dynamic scenarios and achieved significant improvement over recent approaches both in the cases of with-and-without given map of the environment.

1. Introduction

In the last decades application of intelligent robotics systems have increased by many-folds, several domains such as agriculture, manufacturing, surgery, transportation and logistics have been revolutionised by the presence of robotic systems. While classical navigation approaches have enabled robots to autonomously navigate to the desired goal and accomplish the task in hand, sharing same space with humans in diverse environments is still an open research area. Classical approaches often employ a hierarchical strategy towards navigation, a global path planner is used to generate a collision free path to the goal from robot's current position with the help of a given map (Ferguson et al., 2005). This path is then followed by a local planner simultaneously avoiding dynamic obstacles for a fixed horizon. Local path planning approaches for dynamic obstacle avoidance often employ a geometrically motivated optimization strategy that relies on accurate position and velocity estimates of the dynamic obstacles (Rösmann et al., 2015; Fox et al., 1997), which is not always available in practical scenarios. While, recent deep reinforcement learning (DRL) (Everett et al., 2018; Alahi et al., 2016; Liu et al., 2021) have outperformed classical approaches in the field of dynamic obstacle avoidance (map-less navigation), they only consider dynamic obstacles. However, practical scenarios require robots to navigate in both static and dynamic obstacles under partial observability, which poses challenge to these approaches.

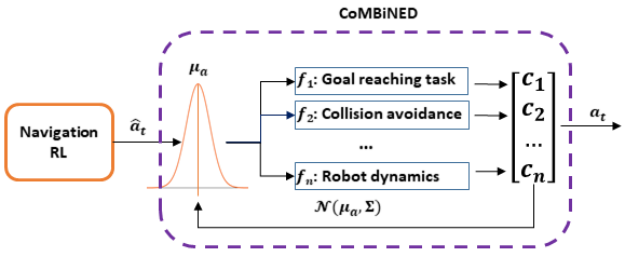

Figure 1. Multi-constraint optimization for navigation: To make safer decisions in complex practical scenarios we divide the task into sub-tasks and learn dynamical models $f_{1:N}$ for them. Our stochastic optimization approach then performs predictive planning at the inference time for the constraints captured by dynamical models for sub-tasks (both analytical and deep Markov networks) in a single framework, which is difficult to achieve by either classical approaches or gradient based approaches. Considering navigation in presence of dynamic obstacles as our primary task we split it into sub-tasks such as (i) goal reaching and (ii) dynamic collision avoidance and plan actions that minimizes the planning costs $c_{1:N}$ for these these sub-tasks which leads to safer actions.

A few recent model-free DRL based approaches (Guldenring et al., 2020; Liu et al., 2020) learn a policy for navigation in dynamic environments directly from range measurements, thus are more suitable to practical scenarios.

[1]Toshiba Research, Cambridge, UK. Correspondence to: Harit Pandya <harit.pandya@crl.toshiba.co.uk>.

Decision Awareness in Reinforcement Learning Workshop at the 39th International Conference on Machine Learning (ICML), Baltimore, Maryland, USA, 2022.

However they suffers from large sim2real gap when subjected to practical scenarios. Motivated from the success of model based reinforcement learning approaches (Ha & Schmidhuber, 2018), Navrep (Dugas et al., 2021) aims to learn the dynamical model first and then use it to learn the policy. However, such a policy tends to be conservative (Dugas et al., 2021), since learning the joint representation that captures the large variations in environment as well as in the behaviors of dynamic obstacles requires training on large amounts of data. Also, performing any further changes to the robot's configuration or accommodating new constraints require retraining of the whole system again.

In this work, we aim to learn a flexible and modular framework for attaining distant point goals in the presence of dynamic obstacles from 2D LiDAR measurements. For circumventing the issues with a joint representation of static and dynamic obstacles, we disentangle the task of navigation into two sub-tasks of: (i) goal reaching and (ii) dynamic collision avoidance. We further train recurrent neural networks for learning the dynamical models for the robot and other obstacle from the data. Considering each task as a constraint, we propose a novel approach called CoMBiNED for multi-constraint planning in deep learning based dynamical systems. Our approach jointly solves the required constraints, analytical or learning based, by computing an optimal action that minimizes the mean expected cost for a planning horizon as shown in Figure 1. To solve the distant horizon task, we also propose a waypoint based planning strategy similar to classical approaches and recent DRL approaches. We maintain a partial occupancy map, which is reconstructed online while navigating. We then select a collision free waypoint on the shortest path, and provide this as a goal to the CoMBiNED.

In summary our contributions are:

- We present a simple yet effective planning framework that can combine several constraints, both deep and analytical, and solves them in continuous space jointly using stochastic optimization at inference time without any retraining of the models.
- Based on our multi-constraint planning framework, we present an indoor navigation approach, that can steer the robot through complex environments in the presence of dynamic obstacles using LiDAR measurements and odometry.
- Our proposed model (CoMBiNED) achieved significant performance gain with respect to the popular dynamical obstacle avoidance methods on the publicly available Navrep benchmark (Dugas et al., 2021).

2. Related Work

Robots co-working along with humans to solve a common goal has been one of the central research theme in robotics and autonomous navigation. Navigation in indoor has always interested robotics research community because of the challenges in localisation, dense clutter and dynamic objects and therefore, it has also been extensively studied. Several classical approaches like time elastic band, dynamic window approach (Rösmann et al., 2015; Fox et al., 1997) have become a standard software stack for robot navigation. Classical approaches divide the navigation task into three stages: (i) a mapper module that identifies the goal and localises current position, (ii) a global planner such as A-star, FMM (Sethian, 1996) that plans a path from current position to the goal (iii) a local planner that takes in a waypoint from the global plan and based on robot's dynamics and obstacle avoidance and plans a trajectory to nearest waypoint. Classical local planners like TEB (Rösmann et al., 2015), DWA (Fox et al., 1997) and MPC (Brito et al., 2019) assume accurate knowledge dynamics of robot as well as other moving obstacles to optimize robot's velocity to attain the waypoint. While classical approaches excel for long horizon point-goal navigation, data driven approaches showcase superior performance on dynamic obstacle avoidance tasks (Kästner et al.). Thus for navigation in dynamical objects, recent approaches either rely on a classical global policy to propose a waypoint (Kästner et al.; Brito et al., 2021) that is followed by a RL based local planner or they switch between classical and DRL planner (Kästner et al., 2021). We showcase through experiments in section 4.4 that simply combining the global planners with RL results in sub-optimal performance, while switching planners could lead to oscillations. Hence, in this work we combine them in a principled fashion using a multi-constrained model based planning.

Model based planning To overcome the limitation of reactive nature of classical planners in complex environments, Model Predictive Control (MPC) based planning has been employed that jointly optimizes the control in presence of both static as well as dynamic obstacles (Brito et al., 2019). Where, the objective is to minimize the goal reaching cost for a fixed horizon, while obstacle avoidance and robot dynamics are treated as constraints. While the approach is able to generate online plans that are inherently safe respecting the constraints, there are a few challenges: (i) Classical MPC solvers do not scale-up to high dimensional inputs such as LiDAR measurements. (ii) Classical MPC based approaches analytically model the dynamics, for highly dynamical systems analytical models could be inaccurate after a short time horizon. Thus, to take the advantage of both MPC and DRL, recent approaches aim to combine MPC's with reinforcement learning. One of the initial approach in this direction was Probabilistic Ensembles with Trajectory

Sampling (PETS) (Chua et al., 2018), which proposed to combine deep networks with planning by propagating samples in the latent space of the learned dynamical model. A recent approach Model-based Policy Planning (POPLIN) (Wang & Ba, 2019), proposed to improve PETS by (i) warm-start of the MPC through policy output which increases the efficiency of planner. (ii) planning in model parameter space instead of action space. In this work, we propose to solve a multi-constraint optimization by formulating each constraint as a model (a neural network or analytic) and employing latent space trajectory sampling to jointly optimize over these constraints. We also use a DRL policy to warm-start the MPC planning thus, making the process more efficient.

The central difference between our approach and existing deep model based planing approaches such as PETS, POPLIN, mu-zero (Schrittwieser et al., 2020) is that: we consider a multi-constraint setting where the objective is to achieve safe and optimal plans over different sub-tasks without retraining the actual policy, on the contrary, PETS/POPLIN optimize over a single task by retraining the current policy. To the best of our knowledge we are the first approach to employ deep model based planning for solving multiple sub-tasks.

In the context of autonomous navigation recent approaches like (Bansal et al., 2020) employ classical MPC to supervise their policy based on imitation learning for visual navigation and (Brito et al., 2019) employ classical MPC for learning better way-points proposal policies for dynamic obstacles avoidance. In this paper, we consider the task of dynamic navigation from LiDAR measurements and odometry. This task is difficult for classical analytical MPC solvers, since LiDAR inputs are high dimensional and do not inherently capture moving obstacles. Thus, we propose recurrent neural networks to model dynamical obstacles and robots's dynamics. We present a novel formulation that can accommodate kino-dynamical models or learned dynamical models as constraints, which are satisfied by online planning in the latent space making our approach inherently safe. To improve the efficiency of online planning we warm start the planning through a model based RL policy and iteratively optimize the samples by maximizing the future expected rewards obtained by propagating the samples through the dynamical models (analytical or learned) of all constraints.

3. Approach

In this section we formulate autonomous navigation as an unconstrained optimization problem (see section 3.1), which is optimally solved by our CoMBiNED framework. Our framework consists of three major components: (i) A recurrent neural network for learning robot's next positions from odometry and LiDAR observations (refer section 3.3). (ii) A spatio-temporal convolutional neural network to model motion of dynamic obstacles from LiDAR observations (as shown in section 3.2). (iii) A multi-constrained model based planning module to find the optimal velocity that is able to steer robot towards goal and is free of collision from dynamic as well as static obstacles (refer section 3.4). Our approach can be summarized by figure 2.

3.1. Problem formulation

We consider the problem of point goal navigation in 2D, where given the current position of the robot $\mathbf{p} = (p_x, p_y)$, and a goal $\mathbf{g} = (g_x, g_y)$ and current LiDAR sensor measurements $\mathbf{x} = (x_1, x_2, ...x_d)$ at the current time instant t, the objective is to navigate the robot to the goal through a series of velocity control commands $\mathbf{a}_t = (a_x^t, a_y^t)$, avoiding N non-cooperative dynamic obstacles and static obstacles defined by a partial-reconstructed map M_t^s. This problem could be formulated as a model predictive control as follows:

$$
\begin{aligned}
\pi^* = & \underset{\pi}{argmin} \sum_{t+1}^{t+H} \|\mathbf{p}_t - \mathbf{g}\| \\
s.t. \quad & \mathbf{p}_{t+1} = f(\mathbf{p}_t, \mathbf{a}_t) \\
& \mathbf{p}_{t+1}^i = f^i(\mathbf{p}_t^i), \quad \forall i \in N \\
& \|\mathbf{p}_t - \mathbf{p}_t^i\| > \delta, \quad \forall i \in N \\
& M_t^s(\mathbf{p}_t) < \epsilon
\end{aligned} \tag{1}
$$

.

Where, π^* is the optimal action sequence that minimizes future distance to the goal for a time horizon H. Here, p_t denotes the position of the robot, p_t^i denotes the position of i^{th} dynamic obstacle and $M_t^s(p_t)$ represents the probability of position p_t to be occupied at a time instance t by static obstacle. We assume that the robot's dynamics evolve by a function f and other dynamic agents evolve through non-stationary dynamics f^i independent of the robot.

Dynamic obstacle avoidance as multi-constraint learning problem: We model our solution as a multi-task planning approach and employ stochastic optimization in latent space to solve all the sub-tasks in a joint framework, thus we call our framework CoMBiNED. There are several advantages in our multi-task planning formulation of the problem: Firstly, we can add additional constraints such as robot's dynamics, smoothness etc. on-the-fly at inference time. Secondly, our framework can handle analytical as well as deep network based dynamic constraints, for example, evolution of robot's trajectory depends upon its dynamics which is an analytic constraint. Whereas trajectory of dynamic obstacles is better predicted by a deep neural network and is difficult to expressed in a closed form. The MPC formulation in equation (1) can be rewritten as an unconstrained optimization

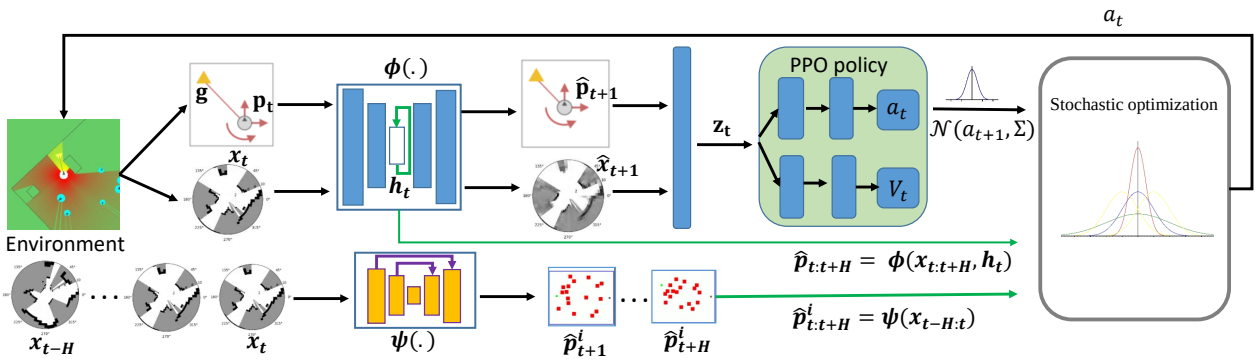

Figure 2. Overall pipeline of the ComBiNED framework: From LiDAR measurement and robot odometry we train two dynamics models for navigation and dynamic obstacle avoidance task. We then solve both these tasks through a stochastic optimization framework for computing the optimal control commands.

problem as,

$$\pi^* = \underset{\pi}{argmin} \sum_{t+1}^{t+H} [C_t], \quad C_t = \alpha C_t^g + \beta C_t^o + \gamma C_t^s \quad (2)$$

Where, we define the goal reaching cost as the geodesic distance between predicted position and goal $C_t^g = d_G(\hat{\mathbf{p}}_t, \mathbf{g})$, which is obtained by computing the length of the shortest path on signed distance field (SDF) representation of current reconstruction of the static map $M_t^s = SLAM(\mathbf{p}_{0:t}, \mathbf{x}_{0:t})$. The predicted position of the robot can be obtained by a recurrent neural network $\hat{\mathbf{p}}_t = \phi(\mathbf{p}_{t-1}, \mathbf{x}_{t-1}, \mathbf{a}_{t-1})$ or using the true dynamics of the robot $f(\cdot)$, if known. Furthermore, we denote the dynamic obstacle avoidance cost C_t^o as the the probability of the robot having collision with a dynamic obstacle. We compute this cost by querying the predicted position of the robot in a dynamic occupancy map i.e. $C_t^o = M_t^o(\hat{\mathbf{p}}_t)$. Where, the dynamic occupancy map is predicted using a spatio-temporal network from past H LiDAR measurements transformed by robot's pose $M_t^o = \psi(\mathbf{p}_{t-H:t}, \mathbf{x}_{t-H:t})$. Similarly, the static collision cost C_t^s is computed from occupancy map (M_t^s) by querying the probability of a position p_t being greater then certain threshold ϵ. Once the future states and corresponding costs are predicted by the recurrent neural networks, we employ a gradient free sampling based stochastic optimization framework to optimize the cost from equation 2. The weights (α, β, γ) of the individual costs are selected as 1, however they can further be tuned.

Occupancy grid mapping It has been reported in literature that the performance of single RL based approaches degrades for long horizon planning (Kästner et al.), (Brito et al., 2021), (Chaplot et al., 2019). Therefore, recent approaches employ a global planner that takes the map and proposes a waypoint near to current position of the robot. In this paper we maintain a partially-reconstructed occupancy map (M_t^s) representation through Simultaneous Localization and Mapping SLAM (Elfes, 1989) from the LiDAR measurements and robot poses. We also maintain a signed distance field (SDF) representation of the world over the partial occupancy reconstruction that is updated after every few steps, since computing SDF is computationally expensive. We then employ fast marching method (Sethian, 1996) as global planner to generate waypoints from the partial SDF after every few steps. As mentioned in the previous sub-sections, this static map is used to compute the collision avoidance cost and the goal reaching cost.

3.2. Modeling Dynamic obstacles

In this paper, we aim to learn the dynamic obstacle trajectories through LiDAR measurements. This is achieved by training a spatio-temporal convolutional neural network, that stacks the Euclidean gridmap representation of previous LiDAR measurements (i.e. $t - H$ to t) and predict dynamic occupancy maps for time interval $t - H$ to $t + H$ which is represented as $M_t^o = \psi(\mathbf{p}_{t-H:t}, \mathbf{x}_{t-H:t})$. Thus, the dynamic collision cost $C_{t+\tau}^o$ can be obtained by probability of a position $(\mathbf{p}_{t+\tau})$ being greater then certain threshold δ. Here, we employ a U-net (Ronneberger et al., 2015) style architecture for modeling ψ, which is trained in a supervised fashion by mapping ground truth locations of dynamic obstacles in same sized grid as LiDAR maps. Note that, In our architectures the channels represent temporally stacked frames thus the convolution operator learns over the spatio-temporal slice of the input data. Hence we call it a spatio-temporal network, alternatively a recurrent neural network could also be used here.

3.3. Robot dynamics prediction

In PETS (Chua et al., 2018), the CEM policy is initialised by random actions, which can be inefficient, since it can

take large number of iterations for CEM to converge. Thus, here we warm start the CEM optimizer by a initial action obtained from a RL policy. To encode the dynamic model of the agent, we employ a state space model that consists of a convolutional varitaional autoencoder (VAE) and a recurrent neural neural network similar to Navrep (Dugas et al., 2021). The LiDAR observations ($\mathbf{x}_t$) are encoded as hidden representation ($\mathbf{z}_t$) by the VAE which is used by a RNN to predict future observations ($\mathbf{x}_{t+1}$). Finally, the hidden representation of the encoder and RNN is stacked to train the PPO (Schulman et al., 2017) RL agent. Once the RL agent is trained is can be employed to generate an initial guess to warm start the CEM process.

Algorithm 1 Multi-constrained model based planning

Input: Constraints as state-space models (RNNs/kino-dynamical), initial guess from RL policy
for L-iterations **do**
  Fit a Gaussian distribution with mean as initial guess from RL and a predefined variance
  Generate N samples from the distribution
  for H-steps **do**
    Propagate samples through **all models**
    Collect the mean expected cost for all samples for a fixed horizon
    Select K-elite samples with minimum costs
    Fit a new Gaussian distribution on them
  end for
end for

3.4. Multi-Constrained model based planning

Since our actions are robot velocities which lie in continuous space, we fit a Normal distribution with the mean as initial solution provided by RL policy, $\widehat{\mathbf{a}}_t = \text{RL}(\mathbf{x}_t, \mathbf{p}_t, \mathbf{g})$ and a predefined variance Σ i.e. $A = \mathcal{N}(\mu = \mathbf{a}_t, \Sigma)$. We then sample N trajectories with jth sample given as $(A^j = [\mathbf{a}^j_{t+1}, \mathbf{a}^j_{t+2}, ..., \mathbf{a}^j_{t+H}])$ from this prior distribution $\mathcal{N}_t$. These samples are propagated from robot's state space model to obtain next states $\widehat{P}^j = \phi(\mathbf{p}^j_t, \mathbf{x}_t, A^j)$, which are then propagated with all the constraint models $(M^o(\widehat{P}^j), M^s(\widehat{P}^j))$ to obtain the mean expected cost C^j for a sample j for all time horizons i.e. $C^j = \sum_{\tau=t+1}^{t=H} C^j_\tau / H$. We then sort these N samples in descending order of the rewards and select K-elite samples with corresponding to minimum C. Following the CEM (Rubinstein, 1999) optimization process, we again fit the normal distribution with means and variance of these K-elite samples. The process is repeated till the mean expect cost does not decrease by a threshold or until a fixed maximum iterations. Finally, we select the best action and command it to the robot in the receding horizon fashion.

4. Experiments

4.1. Network architecture and training

Robot's dynamic model $\phi(\cdot)$: Using the similar architecture to Navrep (Dugas et al., 2021), our VAE for robot's dynamic model (ϕ) consists of a 4-layer convolutional neural network (CNN) encoder, which is then followed by a 2-layer dense layer, and 4-layer CNN decoding blocks. The latent features z are of size 32, and the hidden features of the RNN (we use LSTM as our RNN) are of size 64. The value network of the RL policy comprises of 2-layer dense network.

Trajectory prediction for dynamic obstacles $\psi(\cdot)$: We employ a U-net (Ronneberger et al., 2015) style architecture for our dynamic obstacle branch. The LiDAR measurements are converted to grid maps of size (140×140), 4-grid maps from $(t = t-4 : t = t)$ are stacked in form of $(140 \times 140 \times 4)$ tensor, which is input to the U-net. The U-Net output comprises of $(140 \times 140 \times 8)$ tensor, which represents 8-gridmaps with probability distributions of dynamic obstacles from $(t-4 : t+4)$.

Training data: Similar to Navrep (Dugas et al., 2021), we the dynamic models are trained with 1000 trajectories each of length 200, extracted by controlling the robot using ORCA (Van den Berg et al., 2008) policy for varying number of static and dynamic obstacles between 10 and 15. Furthermore, we also used the leg movement to render dynamic agents as moving legs in the generated LiDAR data and a time-step of 0.2s was used for updating the agents' positions. For training the dynamic obstacles' state space model ($\psi(\cdot)$), we randomly sample 8 consecutive LiDAR maps and give them as input, the true locations of visible dynamic obstacles are mapped in a grid of same size as input and provided to the network.

Training procedure: We use the publicly available pre-trained models provided by Dugas et al. (Dugas et al., 2021) for robot's dynamics and RL policy. We then train the dynamic obstacles' state space models for 200 epochs using the popular Adam (Kingma & Ba, 2014) optimizer with learning late 1e-4. We also used random cropping and random rotations as data augmentations for training the U-net. For CEM optimization we use the horizon length $H = 4$, initial variance 0.8, number of samples = 256 and number of $K-$elite samples = 32, we optimize them for 6 iterations. These hyperparameters of CEM are empirically selected based on Navrep train simulation environment.

4.2. Evaluation on Navrep validation set

While our approach can be applied to any multi-constrained inference problem, here we consider dynamic obstacle avoidance as an application, where the agent needs to reach a point-goal by moving in an obstacle-free space (navigation

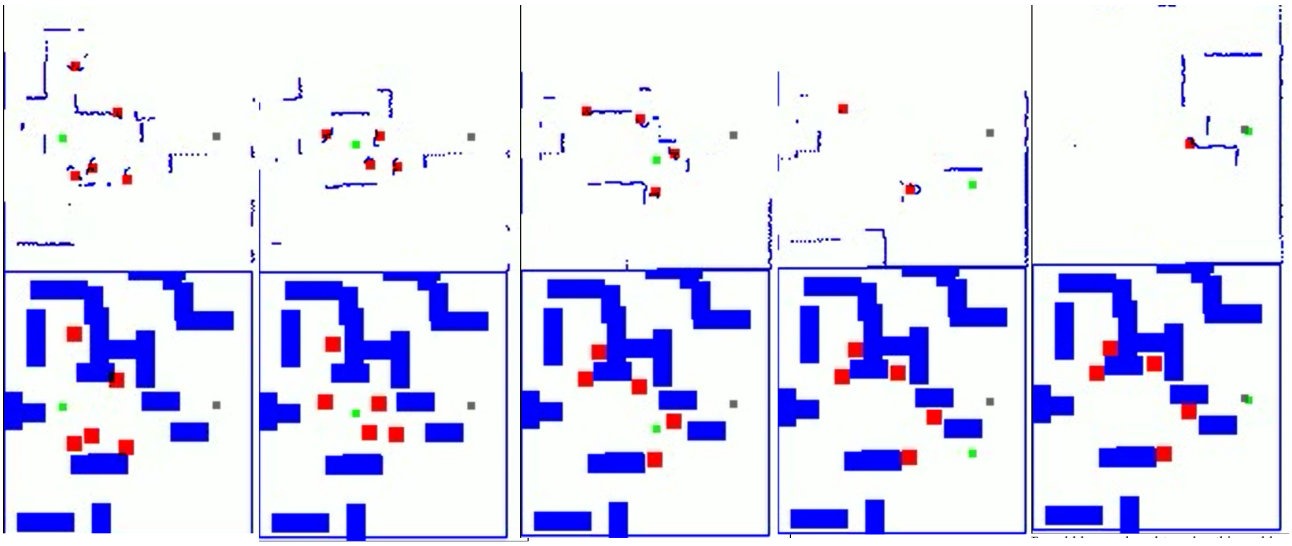

Figure 3. Qualitative result: Example of robot navigating in a complex dynamic scene from our validation set in Navrep simulation environment. (Top) Instantaneous LiDAR reconstructions of the current environment (sparse blue dots) at times t=10,20,30,40 and 50, where robot is green navigates to a goal in grey avoiding dynamic obstacle detections by our dynamic obstacle prediction network in red. (Bottom) Ground truth map. Note that the LiDAR maps are reconstructions of a single 1080 dimensional LiDAR measurements from the sensor, therefore they are sparse.

5 dynamic and 5 static obstacles	Success Rate (%)
Navrep (Dugas et al., 2021)	53
Ours	69

Table 1. Quantitative results on navrep validation set. Here, we achieve 16% improvement success rate over the baseline.

task) and avoid moving obstacles (dynamic obstacle task). We evaluate our approach on our validation environment (5 moving obstacles and 5 objects in a small space) averaged over 100 trajectories. The evaluation results are shown in table 1. It can be seen for the table that our approach outperforms the baseline by a significant margin. Figure 3 shows an example trajectory of our approach where the robot needs to coordinate with other agents due to tight space constraints.

4.3. Quantitative Evaluation on Navrep benchmark

We next evaluate our approach on Navrep's publicly available simulation benchmark without any retraining or finetuning, where our model was tested in 18 unseen scenarios (2 maps 'asl' and 'unity', 9 scenarios each) for 50 times every scenario. The scenario definitions and selection was fixed before any testing took place to avoid bias. A few examples of these scenes are shown in figure 4. It can be seen from 2 that in this benchmark also we outperform Navrep and other RL based baselines. Note that we dropped ('office_j') scene since, the scene was incorrectly rendered in the simulator.

4.4. Effect of using true map

It can further be seen from the table 2, that all the approaches perform significantly better given the true map, this observation validates the myopic nature of the DRL policies. Secondly, it can be also be seen that Navrep when directly combined with global planner performs sub-optimal as compared to us, this showcases the importance CoMBiNED for combining DRL based local planners with classical global planners in a principled fashion.

Approach	**Success(%)**
Navrep (Dugas et al., 2021)	45.5
SOADRL (Liu et al., 2020)	38
Guldenring (Guldenring et al., 2020)	34.3
CoMBiNED (ours)	**51.1**
Navrep (Dugas et al., 2021) + true map	70.1
TEB (Rösmann et al., 2015) + true map	79
CoMBiNED (ours) + true map	**90.4**

Table 2. Quantitative results on navrep benchmark on 18 predefined scenarios. It can be seen that we outperform all the previous approaches in both the cases with-and-without map. Note that even with map and global planner to provide waypoints Navrep (Dugas et al., 2021) performs sub-optimally, which showcases that for combining global planners, the policy needs to readjusted, while CoMBiNED can optimize the policy online.

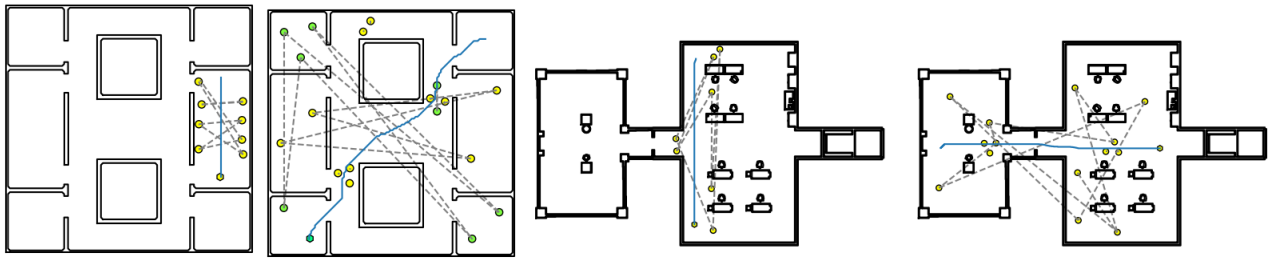

Figure 4. Navrep benchmark: A few example scenarios from Navrep benchmark. Notice the variations in behaviour of dynamic obstacles, few obstacles even remain static. That poses challenge to DRL policies trained only from LiDAR measurements since the policy expects these obstacles to move.

4.5. Planner timing considerations

We evaluate out approach using an Intel-xeon-E5 processor with 128 GB RAM, and a Nvidia-Titan-x GPU, after evaluation we found that for a Navrep planner the policy evaluation time is 0.02 sec., whereas for a non-parallel single core implementation of our CoMBiNED takes 0.2 sec., for 256 samples and 32 k-Elite samples, with horizon length 4 and 6 optimization steps. The advantage of using CEM to solve CoMBiNED is that the process of sampling can be made highly parallel. We observe that the parallel implementation of CoMBiNED takes around 0.08 sec. Thus we firmly believe that CoMBiNED can be implemented on real robotic system.

5. Conclusion

In this paper, we have formulated navigation in dynamic environments as an unconstrained optimization problem and presented a modular online planning approach that can jointly optimize sub-tasks without any modifications or retraining of the deep reinforcement policy. Posing the problem as multi-constraint optimisation makes the approach inherently safe since all constraints are required to be satisfied for a candidate action. Our approach achieved substantially better success rates as compared to recent approaches on publicly available benchmarks for the task of indoor navigation in presence of dynamic obstacles. While in this work, we limit ourselves to the task of indoor navigation, our formulation is generic and could also be applied to even other robotics fields such as manipulation.

References

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., and Savarese, S. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 961–971, 2016.

Bansal, S., Tolani, V., Gupta, S., Malik, J., and Tomlin, C. Combining optimal control and learning for visual navigation in novel environments. In *Conference on Robot Learning*, pp. 420–429. PMLR, 2020.

Brito, B., Floor, B., Ferranti, L., and Alonso-Mora, J. Model predictive contouring control for collision avoidance in unstructured dynamic environments. *IEEE Robotics and Automation Letters*, 4(4):4459–4466, 2019.

Brito, B., Everett, M., How, J. P., and Alonso-Mora, J. Where to go next: learning a subgoal recommendation policy for navigation in dynamic environments. *IEEE Robotics and Automation Letters*, 6(3):4616–4623, 2021.

Chaplot, D. S., Gandhi, D., Gupta, S., Gupta, A., and Salakhutdinov, R. Learning to explore using active neural slam. In *International Conference on Learning Representations*, 2019.

Chua, K., Calandra, R., McAllister, R., and Levine, S. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

Dugas, D., Nieto, J., Siegwart, R., and Chung, J. J. Navrep: Unsupervised representations for reinforcement learning of robot navigation in dynamic human environments. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7829–7835. IEEE, 2021.

Elfes, A. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, 1989.

Everett, M., Chen, Y. F., and How, J. P. Motion planning among dynamic, decision-making agents with deep reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3052–3059. IEEE, 2018.

Ferguson, D., Likhachev, M., and Stentz, A. A guide to heuristic-based path planning. In *Proceedings of the international workshop on planning under uncertainty for autonomous systems, international conference on automated planning and scheduling (ICAPS)*, pp. 9–18, 2005.

Fox, D., Burgard, W., and Thrun, S. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, 1997.

Guldenring, R., Görner, M., Hendrich, N., Jacobsen, N. J., and Zhang, J. Learning local planners for human-aware navigation in indoor environments. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6053–6060. IEEE, 2020.

Ha, D. and Schmidhuber, J. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems 31*, pp. 2451–2463. Curran Associates, Inc., 2018.

Kästner, L., Zhao, X., Buiyan, T., Li, J., Shen, Z., Lambrecht, J., and Marx, C. Connecting deep-reinforcement-learning-based obstacle avoidance with conventional global planners using waypoint generators. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1213–1220. IEEE.

Kästner, L., Cox, J., Buiyan, T., and Lambrecht, J. All-in-one: A drl-based control switch combining state-of-the-art navigation planners. *arXiv e-prints*, pp. arXiv–2109, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Liu, L., Dugas, D., Cesari, G., Siegwart, R., and Dubé, R. Robot navigation in crowded environments using deep reinforcement learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5671–5677. IEEE, 2020.

Liu, Y., Yan, Q., and Alahi, A. Social nce: Contrastive learning of socially-aware motion representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15118–15129, 2021.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Rösmann, C., Hoffmann, F., and Bertram, T. Timed-elastic-bands for time-optimal point-to-point nonlinear model predictive control. In *2015 european control conference (ECC)*, pp. 3352–3357. IEEE, 2015.

Rubinstein, R. The cross-entropy method for combinatorial and continuous optimization. *Methodology and computing in applied probability*, 1(2):127–190, 1999.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609, 2020.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Sethian, J. A. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

Van den Berg, J., Lin, M., and Manocha, D. Reciprocal velocity obstacles for real-time multi-agent navigation. In *2008 IEEE international conference on robotics and automation*, pp. 1928–1935. Ieee, 2008.

Wang, T. and Ba, J. Exploring model-based planning with policy networks. In *International Conference on Learning Representations*, 2019.