Gaussian Processes Incremental Inference for Mobile Robots Dynamic Planning [⋆]

Luka Petrović [*] **Filip Marić** [**] **Ivan Marković** [*] **Ivan Petrović** [*]

[*] *University of Zagreb Faculty of Electrical Engineering and Computing, Laboratory for Autonomous Systems and Mobile Robotics, Croatia (e-mail: name.surname@fer.hr)*
[**] *University of Toronto Institute for Aerospace Studies, Space & Terrestrial Autonomous Robotic Systems Laboratory, Canada*

Abstract: Trajectory optimization methods for motion planning attempt to generate trajectories that minimize a suitable objective function. While such methods efficiently find solutions in static environments, they need to be ran from scratch multiple times in the presence of moving obstacles, which incurs unnecessary computation and slows down execution. In this paper, we propose a trajectory optimization algorithm that anticipates the movement of obstacles and solves the planning problem in an iterative manner. We employ continuous-time Gaussian processes as trajectory representations both for the mobile robot and moving obstacles for which future locations are predicted according to a given model. We formulate the simultaneous moving obstacles tracking and mobile robot motion planning problem as probabilistic inference on a factor graph. Since trajectories of moving obstacles are optimized concurrently to motion planning, the proposed approach works in a predictive manner. After computing the initial solution, we use incremental inference for online replanning after an estimate of the moving obstacle position is provided. Our experimental evaluation demonstrates that the proposed approach supports online motion generation in the presence of moving obstacles.

Keywords: motion planning, trajectory optimization, gaussian processes, factor graphs, incremental inference

1. INTRODUCTION

Motion planning algorithms are necessary for robots that try to navigate through an environment without collisions. They generate trajectories through the robot's configuration space that are feasible and optimal according to some performance criterion, which depends on the robot, task, or environment. Motion planning algorithms that can be executed in real time are of high importance in dynamic environments for fast replanning in response to environment changes.

A significant amount of recent work has focused on trajectory optimization for motion planning. Trajectory optimization methods start with some initial trajectory and then minimize an objective function in order to optimize the trajectory. The CHOMP algorithm by Zucker et al. (2013), utilizes a precomputed signed distance field for fast collision checking and employs covariant gradient descent to minimize obstacle and smoothness costs. An key shortcoming of CHOMP is the need for many trajectory states for reasoning about fine resolution obstacle representations and finding feasible solutions when there are many constraints. Similarly, Schulman et al. (2013) proposed TrajOpt, an algorithm that formulates motion planning as sequential quadratic programming. The key feature of TrajOpt is the ability to solve complex motion planning problems with few states since swept volumes are considered to ensure continuous-time safety. However, if smoothness is required in the output trajectory, post-processing of the trajectory might still be needed thus increasing computation time. In order to overcome the computational cost incurred by using large number of states, Mukadam et al. (2016) and Dong et al. (2016) propose employing Gaussian processes (GP) as continous-time trajectory representations. The proposed Gaussian process motion planning (GPMP2) algorithm formulates the planning problem as probabilistic inference, generating fast solutions by exploiting the sparsity of the underlying linear system. A useful property of GPMP2 is its extensibility and applicability for wide range of problems (Rana et al., 2017; Marić et al., 2019) and in this paper we also rely on the aforementioned framework.

While all of the aforementioned methods efficiently find solutions in static environments, they need to be ran from scratch multiple times in the presence of moving obstacles, which incurs unnecessary computation and slows down the execution. ITOMP, an algorithm that employs incremental trajectory optimization for replanning in dynamic environments, was proposed by Park et al. (2012). ITOMP computes a conservative local bound on the position of each obstacle over a short time and uses that bound to compute a collision-free trajectory for the robot in an incremental manner. However, ITOMP does not predict the motion of dynamic obstacles, thus working in reactive manner.

[⋆] This research has been supported by the European Regional Development Fund under the grant KK.01.1.1.01.0009 (DATACROSS).

In this paper, we propose a trajectory optimization algorithm that anticipates the movement of the obstacles and solves the planning problem in an iterative manner. We employ continuous-time Gaussian processes as trajectory representations both for the mobile robot and moving obstacles for which future whereabouts are predicted according to a given model. We formulate the simultaneous moving obstacles tracking and mobile robot motion planning problem as probabilistic inference on a factor graph. By leveraging tools from the SLAM community, we exploit the structure of the pertaining GPs to find the solution efficiently using numerical optimization. Since moving obstacles trajectories are optimized concurrently to motion planning, the proposed approach works in a predictive manner. After computing the initial solution, we use incremental inference for online replanning after an estimate of the moving obstacle position is provided. We evaluated our method in simulations and compared it to GPMP2 a state-of-the-art trajectory optimization method. A supplementary video of the conducted simulations is available [1]. The results demonstrate that the proposed approach supports online motion generation in the presence of moving obstacles while generating shorter paths and having an order of magnitude faster computation time than GPMP2.

2. TRAJECTORY OPTIMIZATION AS PROBABILISTIC INFERENCE

Following previous work on Gaussian process motion planning and simultaneous trajectory estimation and planning by Dong et al. (2016) and Mukadam et al. (2017a), we view the trajectory estimation and planning problems as probabilistic inference. We seek the *maximum a posteriori* (MAP) continuous-time trajectory given a prior distribution on the space of trajectories and an arbitrary likelihood function.

More formally, we try to find a trajectory $\boldsymbol{x}$ given a collection of events $\boldsymbol{e}$. The posterior density of $\boldsymbol{x}$ given events $\boldsymbol{e}$ can be computed via Bayes' rule from a prior and a likelihood

$$p(\boldsymbol{x}|\boldsymbol{e}) = p(\boldsymbol{x})p(\boldsymbol{e}|\boldsymbol{x})/p(\boldsymbol{e}) \propto p(\boldsymbol{x})p(\boldsymbol{e}|\boldsymbol{x}), \tag{1}$$

where $p(\boldsymbol{x})$ represents the prior on $\boldsymbol{x}$ which encourages smoothness of the trajectory, while $p(\boldsymbol{e}|\boldsymbol{x})$ represents the probability of the events $\boldsymbol{e}$ occurring given $\boldsymbol{x}$. In the following subsections, we define a prior distribution on the space of trajectories and a likelihood function in the context of estimation and planning problems.

2.1 Gaussian process trajectory prior

Consider a continuous-time trajectory as a sample from a vector-valued continuous-time Gaussian process (GP)

$$\boldsymbol{x}(t) \sim \mathcal{GP}(\boldsymbol{\mu}(t), \boldsymbol{\mathcal{K}}(t,t')) \tag{2}$$

that is parameterized with N *support states* at discrete time instants, $\boldsymbol{x}_i \in \mathrm{R}^D$, $i \in N$, where D is the state dimensionality. We employ a structured kernel belonging to a special class of GP priors generated by a linear time-varying stochastic differential equation (LTV-SDE)

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(t)\boldsymbol{x}(t) + \boldsymbol{u}(t) + \boldsymbol{F}(t)\boldsymbol{w}(t), \tag{3}$$

where $\boldsymbol{A}$ and $\boldsymbol{F}$ are system matrices and $\boldsymbol{u}$ is a known exogenous input. The white noise process $\boldsymbol{w}(t)$ is itself a GP with zero mean value

$$\boldsymbol{w}(t) \sim \mathcal{GP}(\boldsymbol{0}, \boldsymbol{Q}_c\delta(t-t')), \tag{4}$$

where $\boldsymbol{Q}_c(t)$ is an isotropic power-spectral density matrix, $\boldsymbol{Q}_c = Q_c\boldsymbol{I}$. A similar dynamical system has been utilized in estimation (Barfoot et al., 2014; Anderson et al., 2015), calibration (Persic et al., 2019) and planning (Mukadam et al., 2017b). The mean and the covariance of the GP generated by the LTV-SDE given in (3) evaluate to

$$\boldsymbol{\mu}(t) = \boldsymbol{\Phi}(t,t_0)\boldsymbol{\mu}_0 + \int_{t_0}^{t} \boldsymbol{\Phi}(t,s)\boldsymbol{u}(s)\,\mathrm{d}s, \tag{5}$$

$$\begin{aligned}\boldsymbol{\mathcal{K}}(t,t') = \boldsymbol{\Phi}(t,t_0)\boldsymbol{\mathcal{K}}_0\boldsymbol{\Phi}(t',t_0)^T + \\ \int_{t_0}^{\min(t,t')} \boldsymbol{\Phi}(t,s)\boldsymbol{F}(s)\boldsymbol{Q}_c(s)\boldsymbol{F}(s)^T\boldsymbol{\Phi}(t',s)^T\,\mathrm{d}s,\end{aligned} \tag{6}$$

where $\boldsymbol{\mu}_0$ and $\boldsymbol{\mathcal{K}}_0$ are the initial mean and covariance of the first state, and $\boldsymbol{\Phi}(t,s)$ is the state transition matrix (Barfoot et al., 2014).

The GP prior distribution is then given in terms of its mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\mathcal{K}}$

$$p(\boldsymbol{x}) \propto \exp\{-\frac{1}{2}\|\boldsymbol{x}-\boldsymbol{\mu}\|^2_{\boldsymbol{\mathcal{K}}}\}. \tag{7}$$

Due to the Markov property of the LTV-SDE in (3), the inverse kernel matrix $\boldsymbol{\mathcal{K}}^{-1}$ is exactly sparse block tridiagonal (Barfoot et al., 2014). A major benefit of modelling continuous-time trajectory in estimation and planning with GPs is the possibility to query the planned state $\boldsymbol{x}(\tau)$ at any time of interest τ, and not only at discrete time instants. In estimation problems, the ability to query the trajectory at any time of interest in a principled way can support seamless integration of asynchronous sensors (Anderson et al., 2015). In planning problems, efficient GP interpolation can be employed for reasoning about small obstacles while keeping a relatively small number of *support states* which reduces the incurred computational burden (Mukadam et al., 2017b). It can also be utilized for providing a dense output trajectory that a robot can execute without any post-processing.

2.2 The likelihood function

The likelihood function encodes information about a particular probabilistic inference problem instance. In estimation problems, the likelihood function encourages posterior trajectories that are in correspondence with measurements, while in planning problems the likelihood function encourages posterior trajectories that are collision-free (Mukadam et al., 2019).

The likelihood function is the conditional distribution $L(\boldsymbol{x};\boldsymbol{e}) \propto p(\boldsymbol{e}|\boldsymbol{x})$ which specifies the likelihood of an event $\boldsymbol{e}$ given trajectory $\boldsymbol{x}$. In the context of motion planning, a binary event corresponds to trajectory state $\boldsymbol{x}_i$ being collision-free, while in the context of estimation an event corresponds to receiving a sensor reading. The likelihood is defined as a distribution in the exponential family (Dong et al., 2016)

$$L(\boldsymbol{x};\mathbf{e}) \propto \exp\left\{-\frac{1}{2}\|\boldsymbol{h}(\boldsymbol{x},\mathbf{e})\|^2_{\boldsymbol{\Sigma}}\right\} \tag{8}$$

[1] youtu.be/1l5ErKT1hmc

where $\boldsymbol{h}(\boldsymbol{x}, \mathbf{e})$ is an arbitrary vector-valued cost function with covariance matrix $\boldsymbol{\Sigma}$.

2.3 Computing the MAP trajectory

The optimal trajectory $\boldsymbol{x}$ is found by maximizing the posterior $p(\boldsymbol{x}|\boldsymbol{e})$ given in (1), using the MAP estimator

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}} \{p(\boldsymbol{x})p(\boldsymbol{e}|\boldsymbol{x})\} = \arg\min_{\boldsymbol{x}} \{\text{-log}(p(\boldsymbol{x})p(\boldsymbol{e}|\boldsymbol{x}))\}. \tag{9}$$

After substituting (7) and (20), the optimal trajectory is

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}} \left\{ \frac{1}{2} \|\boldsymbol{x} - \boldsymbol{\mu}\|^2_{\boldsymbol{\mathcal{K}}} + \frac{1}{2} \|\boldsymbol{h}(\boldsymbol{x}, \mathbf{e})\|^2_{\boldsymbol{\Sigma}} \right\}. \tag{10}$$

The MAP estimation problem can be reduced further to a nonlinear least squares problem which can be solved with iterative optimization approaches such as Gauss-Newton or Levenberg-Marquardt.

3. ITERATIVE INCREMENTAL INFERENCE FOR DYNAMIC MOTION PLANNING

3.1 Probabilistic inference as a factor graph

After solving for the optimal trajectory in (10), we encounter new information during online execution which changes the pertaining likelihood. For example, in estimation problems we encounter new measurements, while in planning problems the environment of the robot can change. A straightforward approach to incorporate this new information would be to resolve the MAP problem repeatedly. However, this can be inefficient and computationally infeasible for an online setting. Mukadam et al. (2019) therefore formulate the probabilistic inference as a factor graph (Kschischang et al., 2001), which is a graphical model that allows exploiting the underlying sparsity of the problem. Afterwards, an incremental inference technique that supports iterative updating of the solution is employed thus increasing computational efficiency.

More formally, the prior and the likelihood functions can be represented as a product of functions that are organized as a bipartite factor graph $G = \{\boldsymbol{X}, \mathcal{F}, \mathcal{E}\}$ (Kschischang et al., 2001)

$$p(\boldsymbol{x})p(\boldsymbol{e}|\boldsymbol{x}) \propto \prod_i f_i(\boldsymbol{X}_i), \tag{11}$$

where variables $\boldsymbol{X} = \{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_N\}$ are a set of robot states along the trajectory, the factors $\mathcal{F} = \{f_0, \ldots, f_M\}$ are functions on variable subsets and $\mathcal{E}$ are edges connected to the two types of nodes. Thus the posterior distribution can be written as a product of the factors that collectively represent the prior and the likelihood

$$p(\boldsymbol{x}|\boldsymbol{e}) \propto f^{prior}(\boldsymbol{X}) f^{like}(\boldsymbol{X}). \tag{12}$$

Barfoot et al. (2014) and Anderson et al. (2015) addressed trajectory estimation problems where the trajectory is represented as a continuous-time GP. The GP representation allows extrapolation thus providing the prediction of future obstacle states, along with estimation. The prior is a joint distribution on the full trajectory $f^{prior} = f^{gp}(\boldsymbol{X})$, while the likelihood represents the probability of all sensor measurements, which factors to $f^{meas} = \prod_i f_i^{meas}(\boldsymbol{X}_i)$. This approach yields a factorization

$$p(\boldsymbol{x}_{est}|\mathbf{e}) \propto f^{gp} f^{meas}. \tag{13}$$

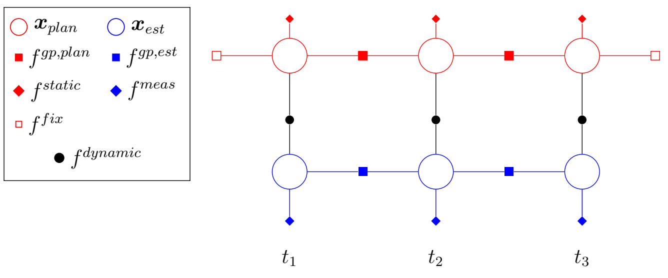

Fig. 1. A simple illustration of a factor graph for simultaneous tracking of a single moving obstacle and mobile robot motion planning with trajectories of three states. The red part of this graph corresponds to the GPMP2 graph given in (16), while the blue part corresponds to the estimation graph given in (13). These two graphs are coupled with a dynamic obstacle factor which allows for predictive motion planning.

Mukadam et al. (2017b) proposed a probabilistic inference framework for solving motion planning problems. Their algorithm, dubbed GPMP2, also utilizes the GP trajectory representation, while the likelihood corresponds to the probability of trajectory being collision-free with obstacles. The likelihood of being collision-free can be factored as

$$f^{obs} = \prod_i f_i^{obs}(\boldsymbol{x}_i). \tag{14}$$

In motion planning problems, there exist a fixed start and goal states which also have to be incorporated into the likelihood. Factors to fix start and goal configurations are therefore employed

$$f^{fix} = f^{start}(\boldsymbol{x}_0) f^{goal}(\boldsymbol{x}_N). \tag{15}$$

The full factor graph of GPMP2 is, therefore

$$p(\boldsymbol{x}_{plan}|\mathbf{e}) \propto f^{gp} f^{obs} f^{fix}. \tag{16}$$

3.2 Simultaneous moving obstacles tracking and motion planning

In real world applications, both estimation and planning problems need to be solved. For example, in dynamic environments, we want to track the moving obstacles and plan the robot's trajectory accordingly. The factor graph of a tracking problem for a single tracked obstacle corresponds to (13), while the factor graph of a planning problem corresponds to (16). The obstacle factor can be split into two factors, one for static and one for dynamic obstacles

$$f^{obs} = f^{static} f^{dynamic}. \tag{17}$$

While the probabilistic inference frameworks presented in Sec. 3.1 can be utilized for solving the moving obstacles trajectory estimation and the robot motion planning independently, we propose solving them concurrently. We formally define the simultaneous moving obstacles tracking and motion planning problem on a factor. The novelty of our method lies in performing inference on a larger factor graph at once, instead of splitting the factor graph into multiple graphs for planning and single obstacle tracking. The full factor graph of the proposed method is

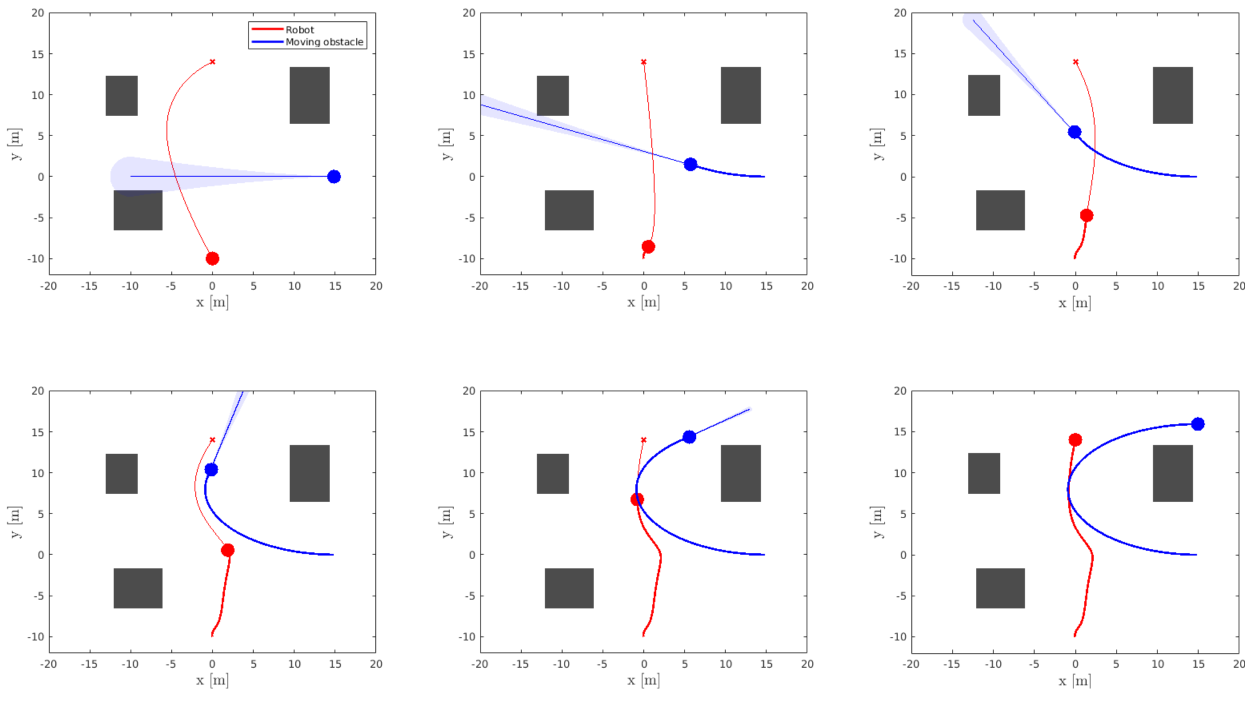

Fig. 2. An example of simultaneous moving obstacle tracking and mobile robot motion planning. The red line represents the robot, with the red circle marking its position, the bold line being the executed trajectory and the thin line representing the currently planned future trajectory. The blue line represents the moving obstacle, with other blue elements corresponding to those in red. Blue shading depicts the covariance of estimation and prediction. The obstacle's trajectory is predicted with uncertainty, and replanning is performed at every timestamp.

$$p\left(\boldsymbol{x}_{plan}, \boldsymbol{x}_{est}^1, \dots, \boldsymbol{x}_{est}^M | \boldsymbol{e}\right) = p(\boldsymbol{x}_{plan}|\boldsymbol{e}) \prod_{m=1}^{M} p(\boldsymbol{x}_{est}^m|\boldsymbol{e}) \tag{18}$$

$$\propto f^{gp,plan} f^{static} f^{fix} \prod_{m=1}^{M} f_m^{gp,est} f_m^{meas} f_m^{dynamic}, \tag{19}$$

where M is the total number of tracked moving obstacles, $f^{gp,plan}$ the GP prior of the robot's trajectory and $f^{gp,est}$ the GP prior of obstacle's trajectory. We solve for the full trajectories of the robot and the moving obstacles simultaneously; the robot's trajectory is planned, while obstacles' past trajectories are estimated and future trajectories predicted. A simple illustration of the proposed factor graph is depicted in Fig. 1. The major advantage of simultaneous optimization is that it allows information flow between the sub-graphs of tracking and planning, which increases performance in both estimation and planning. It gives our planning algorithm its predictive property.

The Gaussian process prior factor is given in (7). We employ measurement, start and goal factors as defined by Mukadam et al. (2019). Dong et al. (2016) define the static obstacles factor which is evaluated using a precomputed signed distance field. Dynamic obstacle factor relies on a hinge loss function of distance between a robot and a tracked dynamic obstacle, similarly to (Petrović et al., 2018), but with covariance of the tracked obstacle used as a parameter

$$f_m^{dynamic} = \exp\left\{-\frac{1}{2}\|\boldsymbol{g}(\boldsymbol{x}_{plan}, \boldsymbol{x}_{est}^m)\|^2_{\boldsymbol{\mathcal{K}}_m}\right\}. \tag{20}$$

3.3 *Iterative incremental inference*

As we discussed in Section 3.1, a major drawback of using nonlinear least squares to solve the MAP inference is that with every update the problem must be completely resolved which is redundant and inefficient. Even if the factor graph is mostly unaltered, the cost on every factor will be evaluated and every variable updated from scratch.

To contend with this problem, Kaess et al. (2012) proposed efficient updates utilizing the Bayes tree data structure, thus we first convert the factor graph into a Bayes tree (Mukadam et al., 2019). When changes in only a few variables or factors are introduced, only the parts of the Bayes tree related with the changes will be updated, while most of the solution will remain the same. Similarly to (Mukadam et al., 2017a), we propose incrementally updating the solution of a factor graph in (19) which significantly improves the efficiency of inference and achieves real-time performance, as we demonstrate in our experiments. The proposed algorithm is summarized in Algorithm 1, while an example of its execution is presented in Fig. 2.

4. EXPERIMENTAL RESULTS

We tested the proposed method on two simulation benchmarks and compared it with the state-of-the-art trajectory optimization technique GPMP2. In Section 4.1, we quantitatively demonstrate the improvement of the proposed method over GPMP2 in scenarios where two dynamic obstacles cross the space between the robot and the goal. In Section 4.2, we demonstrate the improvement of the proposed method over GPMP2 in finding a collision-free

Algorithm 1 Simultaneous moving obstacles tracking and motion planning

```
1:  Initialize trajectories $\boldsymbol{x}_{plan}, \boldsymbol{x}_{est}^{1\ldots M}$
2:  Create factor graph $p\left(\boldsymbol{x}_{plan}, \boldsymbol{x}_{est}^{1}, \ldots, \boldsymbol{x}_{est}^{M} | \boldsymbol{e}\right)$
3:  Perform probabilistic inference to obtain MAP trajectories $\boldsymbol{x}_{plan}^{*}, \boldsymbol{x}_{est}^{*1\ldots M}$
4:  Convert factor graph to a Bayes tree
5:
6:  for $i = 0 \ldots N-1$ do
7:      if $\boldsymbol{x}_{plan,i:i+1}$ is collision-free then
8:          Execute $\boldsymbol{x}_{plan,i:i+1}$
9:          Get moving obstacles' measurements $f^{meas}$
10:         Incremental inference update $\boldsymbol{x}_{plan}^{*}, \boldsymbol{x}_{est}^{*1\ldots M}$
11:     else return failure
12:     end if
13: end for return success
```

trajectory in highly cluttered environment. Both experiments aim to show benefits of the proposed simultaneous optimization for better anticipation of dynamic obstacles and iterative inference for faster replanning.

In our experiments we use the GPMP2 C++ library (Dong et al., 2016, 2017), and its respective MATLAB toolbox, which is based on the GTSAM C++ library (Kaess et al., 2012). Experiments are performed on a system with an Intel Core i7-7700HQ processor and 16 GB of RAM. Both the proposed method and GPMP2 were always initialized with a constant-velocity straight line trajectory. To keep our comparisons fair, in both experiments we used the default parameters from the GPMP2 toolbox both for GPMP2 and for our method. We set the total time for every trajectory as $t_{total} = 10\,\mathrm{s}$ and discretize every trajectory with $N = 100$ states, meaning that the trajectory is replanned every 100 ms. For GPMP2, we incorporate information about dynamic obstacles inside its signed distance field; future trajectory of obstacles is not predicted. For our approach, in both experiments we assume that we measure the positions of obstacles with covariance $\Sigma_{meas} = 0.1$. While our algorithm employs iterative incremental inference, we rerun GPMP2 from scratch at each time step.

4.1 Monte Carlo simulation of crossing agents

The Monte Carlo simulation of crossing agents consisted of 200 synthetic scenarios with randomized initial conditions of moving obstacles. The robot moves from left to right; it starts at position $\boldsymbol{x}_{start} = [-10, 0]\,\mathrm{m}$ and its goal lies at $\boldsymbol{x}_{goal} = [10, 0]\,\mathrm{m}$. Two dynamic obstacles cross the space between the robot and the goal, with one obstacle moving from north to south and the other from south to north. The obstacles move at constant velocity and do not react to the robot's presence. The x coordinates of obstacles' start and goal positions were drawn from a uniform distribution on interval $[-10, 10]\,\mathrm{m}$, while the y coordinates were fixed at $5\,\mathrm{m}$ and $-5\,\mathrm{m}$.

The proposed approach was successful at every scenario and generated paths had the average length of 21.71 m, while GPMP2 was successful in 91.5% of scenarios with the average path length 23.32 m. The histograms of the executed path lengths shown in Fig. 3 demonstrate that our method more often finds direct paths to the goal than

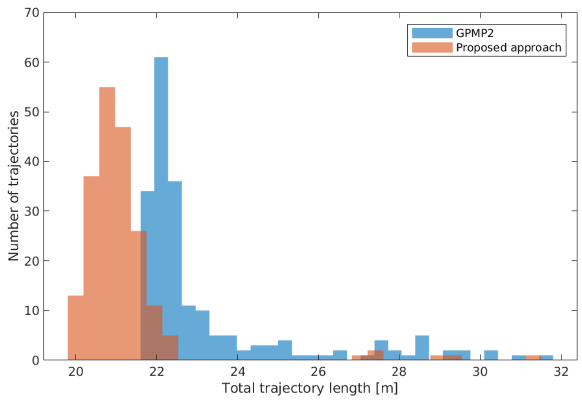

Fig. 3. Histogram of executed path lengths for the proposed approach and GPMP2.

the GPMP2. Due to its predictive property, the proposed approach is better able to navigate tight spaces and replanned paths do not induce significant direction change as is sometimes the case with GPMP2. Each replanning of the trajectory took 5.1 ms with our approach, while it took GPMP2 35.2 ms. An order of magnitude change in computational efficiency is consistent with prior work on incremental inference in similar setting (Mukadam et al., 2017a; Yan et al., 2017). The results demonstrate that iterative incremental inference is suitable for a real-time application, and it can be utilized at significantly higher frequencies compared to running probabilistic inference from scratch at each time step.

4.2 Highly cluttered environment

The highly cluttered environment that we designed for qualitative examination of our algorithm consisted of six static and three dynamic obstacles. The robot's position is again initialized as $\boldsymbol{x}_{start} = [-10, 0]\,\mathrm{m}$ with its goal at $\boldsymbol{x}_{goal} = [10, 0]\,\mathrm{m}$. Every static obstacle is a square with side length $a = 2\,\mathrm{m}$. The static obstacles were initialized at $[5, -1]\,\mathrm{m}$, $[0, 8]\,\mathrm{m}$, $[-7, 4]\,\mathrm{m}$, $[-4, -2]\,\mathrm{m}$, $[4, 4]\,\mathrm{m}$, $[-8, -9]\,\mathrm{m}$, respectively. The three dynamic obstacles were initialized at $[10, 7]\,\mathrm{m}$, $[-5, 8]\,\mathrm{m}$ and $[9, -6]\,\mathrm{m}$.

The proposed approach successfully navigated the robot through this complex environment, as shown in Fig. 4. The executed path length for our approach was 28.48 m, while for GPMP2, which also successfully solved the planning problem, the path length was 34.85 m. Near the first encountered dynamic obstacle, GPMP2 redundantly replanned the trajectory since it had information about robot's path being blocked by an obstacle. This resulted in significantly longer executed path for GPMP2, again demonstrating the benefits of the predictive property of our approach. Each replanning of the trajectory with our approach took 4.6 ms on average, while it took GPMP2 46.8 ms, again showing the computational efficiency of iterative incremental inference compared to recomputing probabilistic inference from scratch.

5. CONCLUSION AND FUTURE WORK

In this paper we have presented a fast trajectory optimization method for motion planning in the presence of moving

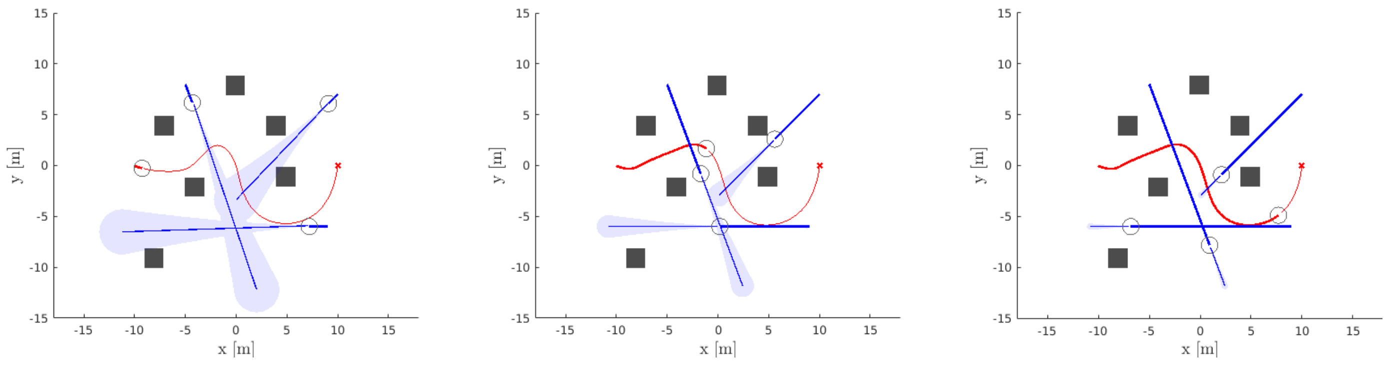

Fig. 4. The resulting trajectory in the highly cluttered environment experiment. Red line represents the robot, where bold line marks executed trajectory and thin line currently planned future trajectory. Blue lines represents the moving obstacles, bold lines mark estimated trajectories and thin lines predicted future trajectories. Blue shading depicts covariance of estimation and prediction.

obstacles. We considered each trajectory as a sample from a continuous-time Gaussian process. We formulated the simultaneous moving obstacles tracking and robot motion planning problem as probabilistic inference on a factor graph. Since moving obstacles' trajectories are optimized concurrently to motion planning, the proposed approach works in a predictive manner. We employed iterative incremental inference for fast replanning suitable for real-time application. We tested our approach in simulation and compared it to GPMP2, a state of the art trajectory optimization algorithm, demonstrating significant improvement in both path lengths and computational efficiency.

In future work, it would be interesting to test the proposed approach in real environments coupling it with a moving obstacles detection and tracking framework. Furthermore, the proposed approach could be extended for higher degree-of-freedom robots, such as mobile manipulators.

REFERENCES

Anderson, S., Barfoot, T.D., Tong, C.H., and Särkkä, S. (2015). Batch nonlinear continuous-time trajectory estimation as exactly sparse gaussian process regression. *Autonomous Robots*, 39(3), 221–238.

Barfoot, T.D., Tong, C.H., and Särkkä, S. (2014). Batch continuous-time trajectory estimation as exactly sparse gaussian process regression. In *Robotics: Science and Systems*, volume 10.

Dong, J., Boots, B., and Dellaert, F. (2017). Sparse gaussian processes for continuous-time trajectory estimation on matrix lie groups. *arXiv preprint arXiv:1705.06020*.

Dong, J., Mukadam, M., Dellaert, F., and Boots, B. (2016). Motion planning as probabilistic inference using gaussian processes and factor graphs. In *Robotics: Science and Systems*, volume 12.

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J.J., and Dellaert, F. (2012). isam2: Incremental smoothing and mapping using the bayes tree. *The International Journal of Robotics Research*, 31(2), 216–235.

Kschischang, F.R., Frey, B.J., and Loeliger, H.A. (2001). Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2), 498–519.

Marić, F., Limoyo, O., Petrović, L., Ablett, T., Petrović, I., and Kelly, J. (2019). Fast manipulability maximization using continuous-time trajectory optimization. *arXiv preprint arXiv:1908.02963*.

Mukadam, M., Dong, J., Dellaert, F., and Boots, B. (2017a). Simultaneous trajectory estimation and planning via probabilistic inference. In *Robotics: Science and Systems*.

Mukadam, M., Dong, J., Dellaert, F., and Boots, B. (2019). Steap: simultaneous trajectory estimation and planning. *Autonomous Robots*, 43(2), 415–434.

Mukadam, M., Dong, J., Yan, X., Dellaert, F., and Boots, B. (2017b). Continuous-time gaussian process motion planning via probabilistic inference. *arXiv preprint arXiv:1707.07383*.

Mukadam, M., Yan, X., and Boots, B. (2016). Gaussian process motion planning. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 9–15.

Park, C., Pan, J., and Manocha, D. (2012). Itomp: Incremental trajectory optimization for real-time replanning in dynamic environments. In *International Conference on Automated Planning and Scheduling*.

Persic, J., Petrovic, L., Markovic, I., and Petrovic, I. (2019). Spatio-temporal multisensor calibration based on gaussian processes moving object tracking. *arXiv preprint arXiv:1904.04187*.

Petrović, L., Marković, I., and Seder, M. (2018). Multi-agent gaussian process motion planning via probabilistic inference. *IFAC-PapersOnLine*, 51(22), 160–165.

Rana, M.A., Mukadam, M., Ahmadzadeh, S.R., Chernova, S., and Boots, B. (2017). Towards robust skill generalization: Unifying learning from demonstration and motion planning. In *Conference on Robot Learning*, 109–118.

Schulman, J., Ho, J., Lee, A.X., Awwal, I., Bradlow, H., and Abbeel, P. (2013). Finding locally optimal, collision-free trajectories with sequential convex optimization. In *Robotics: Science and Systems*, volume 9.

Yan, X., Indelman, V., and Boots, B. (2017). Incremental sparse gp regression for continuous-time trajectory estimation and mapping. *Robotics and Autonomous Systems*, 87, 120–132.

Zucker, M., Ratliff, N., Dragan, A.D., Pivtoraiko, M., Klingensmith, M., Dellin, C.M., Bagnell, J.A., and Srinivasa, S.S. (2013). Chomp: Covariant hamiltonian optimization for motion planning. *The International Journal of Robotics Research*, 32(9-10), 1164–1193.