2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Macau, China, November 4-8, 2019

Fast Run-time Monitoring, Replanning, and Recovery for Safe Autonomous System Operations

Esen Yel and Nicola Bezzo

Abstract— **In this paper, we present a fast run-time monitoring framework for safety assurance during autonomous system operations in uncertain environments. Modern unmanned vehicles rely on periodic sensor measurements for motion planning and control. However, a vehicle may not always be able to obtain its state information due to various reasons such as sensor failures, signal occlusions, and communication problems. To guarantee the safety of a system during these circumstances under the presence of disturbance and noise, we propose a novel fast reachability analysis approach that leverages Gaussian process regression theory to predict future states of the system at run-time. We also propose a self/event-triggered monitoring and replanning approach which leverages our fast reachability scheme to recover the system when needed and replan its trajectory to guarantee safety constraints (i.e., the system will not collide with any obstacles). Our technique is validated both with simulations and experiments on unmanned aerial vehicles case studies in cluttered environments under the effect of unknown wind disturbance at run-time.**

I. Introduction

Autonomous systems with minimal/no supervision are increasingly becoming a reality: autonomous cars and delivery robots are appearing around us especially in major cities, while service and hobby robots are becoming as common as household appliances. As they become more commonplace, it becomes crucial to guarantee safety during their deployment, especially in uncertain environments under the effect of unknown noises and disturbances.

Typical autonomous vehicle operations require the system to follow a trajectory and reach a goal position. However, an autonomous vehicle may not always be able to obtain its state information consistently due to various reasons, including signal occlusions, limited sensor capabilities, and sensor failures. For instance, an aerial vehicle may lose its GPS signal while flying in between tall buildings or under trees and may not be able to obtain its position information unless it is above a certain altitude. In such cases, since measurements are not available, the vehicle is not capable of adapting its behavior according to its current state, which may lead to unsafe states (e.g., collision with obstacles) due to uncertainties and disturbances. Thus, it is necessary to create proactive systems capable of predicting and assessing future states and replan accordingly to avoid possible unsafe conditions in the future. Reachability analysis is a well-known approach to deal with such a problem and estimate future states of a system under uncertainties; however, this process can be computationally complex and thus not suitable for run-time applications.

To estimate the future states of a system at run-time and fast, we leverage Gaussian process (GP) theory to obtain a regression that is used online to estimate the maximum deviation for new trajectories. Guarantees for this reachable set estimation approach are presented by showing that the actual deviation is less than its GP-based estimation. By using the proposed method, the system becomes capable of predicting and assessing its future positions and taking actions in a timely manner to guarantee safety (i.e., avoid collision with any obstacle) and liveness (i.e., follow the desired trajectory closely) when measurements are intermittent. The proposed self/event-triggered monitoring, recovery, and replanning framework schedules safe recovery maneuvers to obtain state information and guarantee safety, and replans the trajectory from the observed states whenever deemed necessary. Fig. 1 shows a pictorial representation of the proposed framework in which a quadrotor builds reachable sets at run-time fast to estimate future states that it could cover in GPS denied environments with sporadic pose observations, also predicting recovery maneuvers whenever needed if the reachable set intersects with any obstacle and no observations are available before entering an unsafe state.

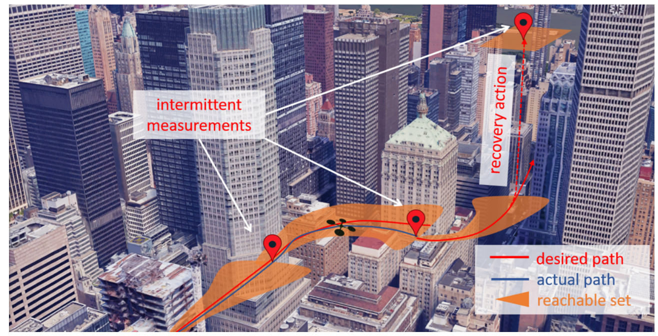


Fig. 1. Pictorial representation of the envisioned fast monitoring, replanning, and recovery approach in which a UAV computes fast reachable sets during run-time and predict recovery and replanning actions when necessary.

To summarize, we aim to solve the following challenges:

- how to perform fast reachable set estimation online and assure that the system will be inside these sets;
- how to guarantee safety and liveness properties when observations are intermittent in the presence of noise and unknown disturbance effects in the environment.

The contribution of this work is twofold: 1) we introduce a novel Gaussian process-based approach to predict the future states of the system fast at run-time, 2) we propose a self/event-triggered framework for monitoring, recovery, and replanning of an autonomous system to guarantee safety and liveness conditions.

Throughout the paper, we consider a navigation case study with an unmanned aerial vehicle (UAV) aiming to reach a goal position in a cluttered environment. The environment can prevent the system from being able to observe its state information constantly (e.g., because of trees or tall buildings). The system can obtain its position information at random times and can perform recovery actions (e.g., change

Esen Yel and Nicola Bezzo are with the Departments of Systems and Information Engineering and Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904, USA {esenyel, nbezzo}@virginia.edu

978-1-7281-4004-9/19/$31.00 ©2019 IEEE

its height) to get measurements when necessary.

The rest of the paper is organized as follows: in Section II we summarize the related work and in Section III, we formally define the problem. In Section IV, we present our proposed approach, which is validated with simulations and experiments in Sections V and VI, respectively. Finally, we draw conclusions and discuss future work in Section VII.

II. Related Work

Reachability analysis is a common approach used in robotics to compute the possible future states of a system and to guarantee safety. Hamilton-Jacobi (HJ) reachability analysis is a widely used reachability approach for safety guarantees of optimal system trajectories for hybrid systems [1]. HJ reachability analysis offers flexibility by considering different system dynamics, uncertainties, and control policies, however, it suffers from computational scalability [2]. To deal with stochastic dynamics, Fourier transforms have been used to perform stochastic reachability for linear systems in [3], and to plan in dynamic environments [4]. In [5], SOS programming was used in order to numerically calculate funnels, which are analogous to reachable tubes. Matrix measures were used in [6], [7] to provide bounds on the divergence from the trajectories, and hence were utilized for reachability analysis.

Because the analytical calculation of reachable sets for nonlinear systems with complex dynamics is computationally expensive, the robotics community has recently started to use machine learning to speed up the process. In [8], the authors defined the reachability problem as an optimization problem and compared linear regression and support vector machine (SVM) methods to compute the optimal cost to make reachability classifications. In [2], neural networks were proposed to be used to approximate HJ reachability in a computationally efficient way. In this work, we utilize Gaussian Processes (GP) regression to estimate the reachable sets of the system. GP regression is widely used in robotics, for example, to learn and cancel the modeling error in model reference adaptive control (MRAC) [9] or for model learning as demonstrated in [10]. Different from these works, here, we use GP regression to provide bounds to the maximum deviation of the system from its desired trajectory based on the time to complete the planned trajectory. In our previous work [11], we introduced an adaptive reachability-based self-triggered scheduling and replanning policy to minimize the sensor monitoring operations for a UAV in a cluttered environment under the effect of external disturbances and system noises. In this work, we build on this previous work and propose a self/event-triggered scheduling and planning approach to guarantee the safety and liveness of a system with intermittent state information, noise, and disturbance.

III. Problem Formulation

In this paper, we are interested in finding a technique for fast reachability to monitor the state of a system and recover and replan accordingly to guarantee both safety and liveness properties. Formally:

Problem 1: ***Fast Reachability*:** A UAV has the objective to follow an obstacle-free trajectory in a cluttered environment with intermittent state measurement under disturbances. Given the UAV dynamics $\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{w}(t))$ as a function of its state $\boldsymbol{x}$, input $\boldsymbol{u}$, and disturbance $\boldsymbol{w}$, find a policy to quickly estimate the reachable sets $\boldsymbol{R}(\boldsymbol{x}_\tau, t)$ of the system at time t while tracking a desired trajectory $\boldsymbol{x}_\tau$ under the effect of unknown disturbance, measurement and input noises during run-time. Both the disturbance and noise values are assumed to be bounded in magnitude but unknown at run-time.

Problem 2: ***Self/Event-triggered Monitoring, Recovery and Replanning*:** Once reachable sets are obtained by solving Problem 1, find an online policy to schedule the time in which the system needs to switch to a safe recovery mode to observe its state, and to replan its trajectory in order to satisfy the following safety and liveness conditions:

- *Safety Constraint:* The UAV should avoid collisions with obstacles:

$$\|\boldsymbol{p}(t) - \boldsymbol{p}_{oi}\| > r_{oi}, \forall t \in [t_p, t_p + T], \forall i \in [0, N_o] \quad (1)$$

  in which $\boldsymbol{p}(t) = [x, y]^\mathsf{T}$ is the position of the vehicle, $\boldsymbol{p}_{oi} = [x_{oi}, y_{oi}]^\mathsf{T}$ is the position of the i^{th} obstacle in the $x - y$ plane and r_{oi} is its radius, N_o is the number of obstacles in the environment, t_p is the planning time of the operation, and T is the duration of the trajectory. Obstacles are assumed to have circular shapes.
- *Liveness Constraint:* The UAV should stay within a certain proximity of the planned trajectory:

$$\|\boldsymbol{p}(t) - \boldsymbol{p}_\tau(t)\| \leq \lambda_d, \forall t \in [t_p, t_p + T] \quad (2)$$

  where $\boldsymbol{p}(t)$ and $\boldsymbol{p}_\tau(t)$ are the actual and desired positions of the vehicle along the trajectory at time t respectively, and λ_d is the allowed deviation threshold.

IV. Fast Run-time Monitoring, Recovery and Replanning

In this section, we describe our framework for fast reachability analysis and online monitoring, recovery, and replanning. The proposed architecture consists of offline and online stages, as displayed in Fig. 2.

We estimate reachable sets online by using a novel Gaussian Process (GP)-based regression technique. Training for GP regression model is performed on a library of trajectory primitives with different durations. These primitives are run on the UAV offline under various disturbances, and their corresponding maximum deviation values are also saved in the library. At run-time, an obstacle-free trajectory is planned from a given initial state to a goal state. The trained GP regression model is then used to estimate the maximum deviation of the vehicle for this new trajectory.

Because the state measurement is assumed to be intermittently available as discussed in Section I, the UAV may not be capable of observing its position information constantly. Instead, when measurements are missing, the system will use its model to calculate its position considering ideal conditions (i.e., no disturbance and noise).

The task here is to estimate the maximum deviation over time (i.e., the position reachable tube) due to noise and disturbances during the interval of times in which the system is running without receiving updated measurements. Based on the estimated position reachable tube, a self-triggered monitor is deployed to compute the first time that a system may violate the safety constraint under the worst-case assumption that it will never be able to obtain its position information. At that time, the UAV will need to recover its position

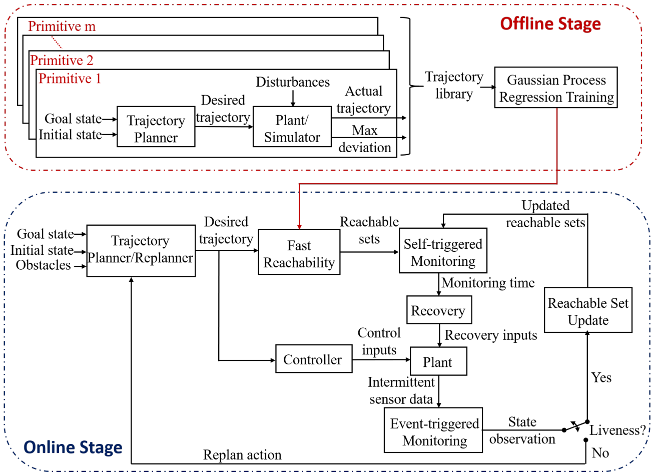


Fig. 2. Architecture of the proposed approach.

information by switching into a predefined safe maneuver, which is defined as moving to an altitude above a certain level in this work. In general, recovery may not always be feasible, and it is subject to disturbance and uncertainties as well. Therefore, during planning, the time required to perform the recovery operation needs to be considered too. Once a recovery operation is completed, the trajectory is replanned using the obtained state information. If the position information becomes available before the predicted unsafe time (i.e., GPS information might be unavailable or available depending on the position of trees, buildings, etc.), an event-triggered procedure is deployed to assess the current state of the vehicle. If the vehicle is within the close proximity of the desired trajectory, the reachable sets are shrunk depending on the deviation. If the vehicle deviates too much from the desired trajectory, the trajectory is replanned for liveness.

The first step in our approach consists in using Gaussian processes theory to estimate reachable sets, which will be explained in detail in the following section.

A. Gaussian Process-based Fast Reachability

Given the premises in the previous section, the position reachable set at time t when the UAV is running without state measurement is defined as:

$$\boldsymbol{R}(\boldsymbol{p}_\tau, t) = \{\boldsymbol{p}(t) : \|\boldsymbol{p}(t) - \boldsymbol{p}_\tau(t)\| \leq d_m(t)\} \qquad (3)$$

where $d_m(t)$ is the maximum deviation at time t from the desired trajectory position $\boldsymbol{p}_\tau$.

In order to estimate the position reachable sets fast and avoid computationally expensive traditional reachability analysis tools [11], [12], we leverage GP regression to estimate $d_m(t)$ based on a library of previously collected trajectory primitives data.

1) Training Data Collection: GP regression training is obtained by calculating and running primitive trajectories offline and by generating a library of trajectories of different duration. The trajectory library should be rich enough in terms of the duration, length, initial, and final velocities to model the maximum deviation behavior. For this reason, we generated trajectories starting from the origin with various initial velocities to different goal locations using minimum jerk trajectory generation [13]. These trajectories were executed using a PID controller without position measurements under a rich wind disturbance set during training. The wind disturbances inside the disturbance set $\mathcal{W}$ are in four main directions with constant or sinusoidal magnitudes with different frequencies, and they are assumed not to change direction during the operation. During the online stage, the disturbance has unknown direction and magnitude; however, its magnitude is assumed to be bounded by the maximum magnitude value in $\mathcal{W}$. The PID controller is assumed to remain the same during the offline and online stages.

A set of sample trajectories from the UAV trajectory primitive library can be seen in Fig. 3. The red curves correspond to the desired trajectory, and the blue curves correspond to the actual trajectories. In total, we collected 1250 different trajectory primitives, which are assumed to be representative enough to model the maximum deviation behavior at runtime as they capture a wide range of durations, initial and final velocities and lengths.

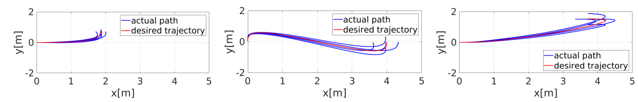
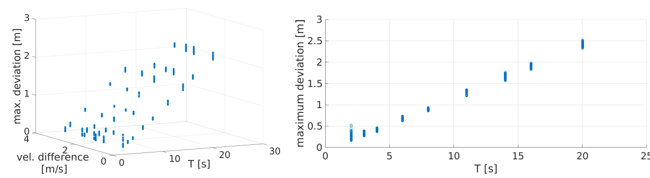


Fig. 3. Example trajectories from the trajectory primitive library.

To understand which factors affect the maximum deviation from the desired trajectory, we have analyzed the relationship between the maximum deviation and various trajectory specifications such as the difference between initial and final states, average velocity, trajectory length, and duration. For example, consider the plot in Fig. 4(a). As can be noticed, there is no correlation between the magnitude of the maximum deviation from the desired trajectory and the difference between initial and final velocities. On the contrary, the time duration of the trajectories has a definite impact on the maximum deviation (Fig. 4(b)), and therefore, it is chosen as the regression variable. The results of this analysis were expected since deviation increases when a system is exposed to a disturbance for a longer time, especially in open loop.



(a) (b)

Fig. 4. (a) Maximum deviation as a function of time (i.e.,duration of the trajectory) and difference between initial and final velocities. (b) Maximum deviation as a function of time.

For a given trajectory of duration T, the maximum deviation from the desired trajectory is calculated as follows:

$$d_M(T) = \max_{\boldsymbol{w} \in \mathcal{W}} \max_{t \in [0,T]} \|\boldsymbol{p}_{\boldsymbol{w}}(t) - \boldsymbol{p}_\tau(t)\| \qquad (4)$$

where $\boldsymbol{p}_{\boldsymbol{w}}(t)$ is the position of the vehicle at time t which is following the trajectory $\boldsymbol{p}_\tau(t)$ under the disturbance $\boldsymbol{w} \in \mathcal{W}$.

For each primitive trajectory, the corresponding duration and maximum deviation values are stored in a trajectory primitive library. For a library consisting on m different trajectories, the trajectory durations are saved in a vector $\boldsymbol{t} = [T_1, T_2, \cdots, T_m]$ and the corresponding maximum deviation values are saved in a vector $\mathbf{d}_M = [d_M(T_1), d_M(T_2), \cdots, d_M(T_m)]$. Given this trajectory primitive library, Gaussian process regression is utilized in the following subsection to estimate the maximum deviation of a new trajectory.

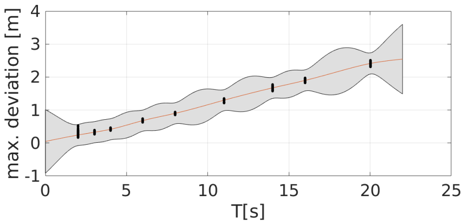


Fig. 5. GP regression of maximum deviation as a function of the trajectory duration implemented on Matlab using GPML toolbox [15].

2) Gaussian Process Regression: Gaussian process regression is a nonparametric regression technique which is used in this work to find a mapping between trajectory duration T and maximum deviation $d_M(T)$. Given a trajectory library containing the set of collected observations $\mathcal{D} = \{\mathbf{t}, \mathbf{d}_M\}$, our goal is to predict the maximum deviation for a new input $\boldsymbol{t}^*$ by drawing $\boldsymbol{d}_M^*$ from the posterior distribution $p(\boldsymbol{d}_M^*|\mathcal{D})$. By definition of GP [14], previous observations $\boldsymbol{d}_M$ and function values $\boldsymbol{d}_M^*$ follow a joint (multivariate) normal distribution:

$$\begin{bmatrix} \boldsymbol{d}_M \\ \boldsymbol{d}_M^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{t}) \\ \mu(\boldsymbol{t}_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I} & \boldsymbol{K}_* \\ \boldsymbol{K}_*^T & \boldsymbol{K}_{**} \end{bmatrix} \right) \quad (5)$$

where $\boldsymbol{K} \in \mathbb{R}^{m \times m}$ has entries $\boldsymbol{K}_{(i,j)} = k(t_i, t_j)$ for $i, j \in \{1, \cdots, m\}$, $\boldsymbol{K}_* = [\, k(t_1, \boldsymbol{t}^*) \ \cdots \ k(t_m, \boldsymbol{t}^*) \,] \in \mathbb{R}^{m \times m_*}$ and $\boldsymbol{K}_{**} = k(\mathbf{t}_*, \mathbf{t}_*) \in \mathbb{R}^{m_* \times m_*}$. σ_ϵ^2 is the noise level associated with the observations, m is the size of the observation set, m_* is the size of the test set, μ is the mean function and k is the covariance function. We chose to use the widely known Matern kernel [14] as a covariance function.

The estimation of $\boldsymbol{d}_M^*$ conditioned on the observations $\mathcal{D}$ is calculated using the properties of joint Gaussian distributions. Namely, the posterior probability is also a Gaussian distribution:

$$p(\boldsymbol{d}_M^*|\boldsymbol{t}_*, \boldsymbol{t}, \boldsymbol{d}_M) \sim N(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}_*) \quad (6)$$

with the following mean and covariance:

$$\boldsymbol{\mu}^* = \mu(\boldsymbol{t}_*) + \boldsymbol{K}_*^T(\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I})^{-1}(\boldsymbol{d}_M - \mu(\boldsymbol{t})) \quad (7)$$

$$\boldsymbol{\Sigma}_* = \boldsymbol{K}_{**} - \boldsymbol{K}_*^T(\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I})^{-1}\boldsymbol{K}_* \quad (8)$$

In Fig. 5, the GP regression model learned using our trajectory primitive library is shown. Learning is performed using 1250 data points shown by black dots in Fig. 5, and regression is performed on 1000 time duration data points with values between 0 and 22 seconds. The red curve is the mean for the data points and the shaded gray region is the 95% confidence interval. In order to verify this regression, we performed a test on 110 untrained trajectories and we observed that all of them had smaller maximum deviation values than the upper bound of the confidence interval.

At run-time, given a new trajectory of duration T^*, the maximum deviation estimation is finally calculated as:

$$\tilde{d}_M(T^*) = \mu^* + 2\sigma_* \quad (9)$$

which corresponds to the upper bound of the 95% confidence interval where μ^* and σ_*^2 are calculated according to (7) and (8) respectively for $\boldsymbol{t}_* = T^*$.

From the collected training data, we observed that the maximum deviation from the trajectory grows linearly over time, as can be noted in Fig. 6. We have also observed that the initial overshoot in the maximum deviation is due to the aggressiveness of some trajectories. For ease of discussion in this work, we are neglecting this behavior, but it can be easily included in our model by setting an initial deviation offset. Based on this linear dependency, the maximum deviation at time t is calculated as follows:

$$d_m(t) = \tilde{d}_M(T^*)\frac{t}{T^*} \quad (10)$$

The reachable set $\boldsymbol{R}(\boldsymbol{p}_\tau, t)$ for a trajectory of duration T^* is finally obtained according to (3) with $d_m(t)$ in (10).

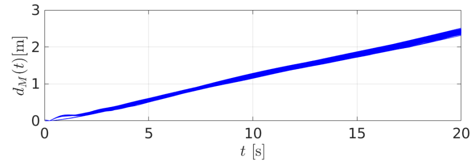


Fig. 6. Maximum deviation values over time for trajectories with 20 seconds duration under the effect of a range of disturbances.

GP regression can be used to calculate reachable sets because the maximum deviation from the desired trajectory is bounded by its estimated value $\tilde{d}_M(T^*)$ if the training set includes the maximum possible deviations as will be shown in Lemma 1 and 2:

Lemma 1: For two trajectories in the training set of consecutive durations T_1 and T_2 respectively, the difference between their maximum possible deviation values $|d_M(T_1) - d_M(T_2)|$ is bounded if the difference between their time duration is also bounded: $|T_1 - T_2| \leq \epsilon$.

Proof: The system state without the effect of disturbance $\hat{\boldsymbol{x}}(t)$ and under disturbance $\boldsymbol{x}(t)$ evolves over time as follows:

$$\dot{\hat{\boldsymbol{x}}}(t) = g(\hat{\boldsymbol{x}}(t), \boldsymbol{u}(t)) \quad (11)$$

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{w}(t)) = g(\boldsymbol{x}(t), \boldsymbol{u}(t)) + \boldsymbol{G}\boldsymbol{w}(t) \quad (12)$$

where $\boldsymbol{u}(t)$ is the controller input, $\boldsymbol{w}(t)$ is the bounded external disturbance: $\|\boldsymbol{w}(t)\| \leq W$ and $\boldsymbol{G}$ is the disturbance matrix [16]. The initial conditions for both systems are the same: $\boldsymbol{x}(0) = \hat{\boldsymbol{x}}(0) = \boldsymbol{x}_0$. From (12) the disturbance effect is additive. When there is no state measurement, the system generates the control inputs with the assumption of ideal conditions (i.e., no disturbance and noise), hence, the same input $\boldsymbol{u}(t)$ is applied to both systems in (11) and (12).

The difference between the systems states with and without the effect of disturbance at the end of a trajectory is calculated based on the difference between the dynamics of the two systems in (11) and (12):

$$\hat{\boldsymbol{x}}(T_1) - \boldsymbol{x}(T_1) = \int_0^{T_1} \boldsymbol{G}\boldsymbol{w}(\tau)d\tau \quad (13)$$

Since the disturbance is bounded, the norm of the difference between the actual and the nominal state is also bounded at the end of the trajectory.

$$\|\hat{\boldsymbol{x}}(T_1) - \boldsymbol{x}(T_1)\| \leq T_1 W_G \quad (14)$$

where W_G is the upper bound of the effect of the disturbance: $\|\boldsymbol{G}\boldsymbol{w}(t)\| \leq W_G, \forall t \in [0, T]$.

Thanks to the stability of the system [13], the deviation between the desired state (desired trajectory) and the system state without the effect of disturbance is bounded at the end of the trajectory when there is no disturbance in the environment:

$$\|\hat{\boldsymbol{x}}(T_1) - \boldsymbol{x}_\tau(T_1)\| \leq \xi \quad (15)$$

Using (15) and (14), it can be shown that the distance between the actual state and the desired state is bounded at the end of the trajectory:

$$\|\boldsymbol{x}(T_1) - \boldsymbol{x}_\tau(T_1)\| = d(T_1) \leq d_M(T_1) = \xi + T_1 W_G \quad (16)$$

For a trajectory of duration $T_2 = T_1 + \epsilon$ where ϵ is a positive real number, the deviation is bounded as follows:

$$\|\boldsymbol{x}(T_2) - \boldsymbol{x}_\tau(T_2)\| = d(T_2) \leq d_M(T_2) = \xi + (T_1 + \epsilon) W_G \quad (17)$$

Therefore, if the difference in the durations of two different trajectories is bounded, the difference between their maximum possible deviation values is also bounded:

$$d_M(T_2) - d_M(T_1) = \epsilon W_G \quad (18)$$

■

Lemma 2: Given a trajectory, the GP regression estimation for the maximum deviation value $\tilde{d}_M$ calculated as in (9) is an upper bound to the actual maximum deviation d_M from that trajectory.

Proof: Let's consider the data points in the training set with maximum deviations $d_M(T_1) = \xi + T_1 W_G$ and $d_M(T_2) = \xi + T_2 W_G$. Based on the system dynamics and (16), the deviation at time $T^* = T_1 + \varepsilon < T_2$ is bounded: $d(T^*) \leq d_M(T^*) = \xi + (T_1 + \varepsilon) W_G$ where $\varepsilon > 0$.

The slope of the virtual line which connects these two data points is: $\frac{d_M(T_2) - d_M(T_1)}{T_2 - T_1}$ and the deviation value on this line for T^* can be calculated as follows:

$$\begin{aligned}\bar{d}_M(T^*) &= d_M(T_1) + \frac{T^* - T_1}{T_2 - T_1}(d_M(T_2) - d_M(T_1)) \\ &= d_M(T_1) + \frac{\varepsilon}{T_2 - T_1}(T_2 - T_1) W_G = d_M(T_1) + \varepsilon W_G\end{aligned}$$

For a GP regression with the upper bound of the confidence interval above this virtual line, the maximum deviation estimation is always larger than the actual maximum deviation:

$$\tilde{d}_M(T^*) \geq \bar{d}_M(T^*) = d_M(T_1) + \varepsilon W_G = d_M(T^*) \quad (19)$$

■

As shown in the previous lemma, if the training set contains the maximum possible deviations for the given trajectory durations, GP-based maximum deviation estimation can provide upper bounds to the actual deviation for any type of stable system. With rich enough training sets, these upper bounds can be tight whereas lack of enough training data would cause over-conservative estimations. In any case, the estimated reachable sets contain all the states that the system can actually reach under bounded disturbance. Therefore, these estimated reachable sets can be used to guarantee safety as will be explained in the following section.

B. Self/Event-triggered Monitoring, Recovery, and Replanning

With the ability to estimate reachable sets at run-time, we design a policy for online monitoring to assess how long the system could continue its motion without a position measurement and for scheduling the recovery which allows the vehicle to obtain its position information. The system needs to monitor its position before a collision may occur. The earliest time that a collision may occur, here referred to as *monitoring time*, is computed as:

$$t_{s+1} = \min(t_k | \boldsymbol{R}(\boldsymbol{p}_\tau, t_k \in [t_p, t_p + T]) \cap \boldsymbol{O} \neq \emptyset) - t_r \quad (20)$$

where t_r is the amount of time necessary for a safe recovery maneuver. At t_{s+1}, the system switches to a recovery operation which allows it to observe its position information (e.g., by flying above a building). After recovering the position information, the system replans its trajectory accordingly as represented in Fig. 2.

After the start of the operation until the recovery maneuver, the position information may become available at random times $t_m \in [t_p, t_{s+1}]$ which are unknown a priori. According to the obtained position information, two procedures (explained next) are implemented on the vehicle: either 1) reachable set shrinking if the deviation from the desired trajectory is less than a predefined liveness threshold λ_d, or 2) trajectory replanning if the deviation is above λ_d.

1) Reachable Set Shrinking: Reachable sets are estimated with the worst case scenario assumptions in terms of disturbance and noise. Therefore, when the disturbance in the environment is not as strong as its maximum value, the reachable set estimation might become over-conservative. To have more accurate estimations about where the system could reach and farther reduce computation, we leverage the deviation from the desired trajectory at the time in which the position information becomes available and shrink the reachable sets [17] without the need to compute a new reachable set. The deviation from the desired trajectory at monitoring time is calculated as follows:

$$d(t_m) = \|\boldsymbol{p}(t_m) - \boldsymbol{p}_\tau(t_m)\| \quad (21)$$

where $\boldsymbol{p}(t_m)$ is the position of the UAV at time t_m and $\boldsymbol{p}_\tau(t_m)$ is the desired position on the planned trajectory at t_m. If $d(t_m) \leq \lambda_d$, the reachable sets are updated as follows:

$$\begin{aligned}&\boldsymbol{R}'(\boldsymbol{p}_\tau, t) = \{\boldsymbol{p}(t) : \|\boldsymbol{p}(t) - \boldsymbol{p}_\tau(t)\| \leq (d_m(t) - d(t_m))\} \\ &\text{with } \ t \in [t_m, t_p + T]\end{aligned} \quad (22)$$

Fig. 7 displays the reachable set shrinking procedure where the original position reachable tube $\boldsymbol{R}(\boldsymbol{p}_\tau, t \in [t_p, t_p + T])$ is shown by a green region and the shrunk position reachable tube $\boldsymbol{R}'(\boldsymbol{p}_\tau, t \in [t_m, t_p + T])$ is displayed by a dark orange region. As can be seen, the reachable sets become smaller since the deviation from the desired trajectory is smaller than the estimated maximum deviation value.

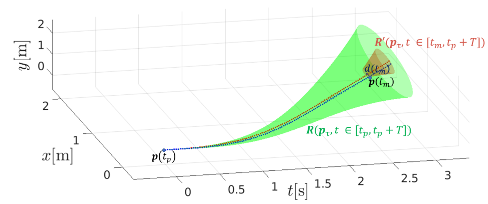


Fig. 7. Updating the reachable tubes based on the deviation from the desired trajectory.

2) Event-triggered Replanning: When the system state information is obtained, the UAV may decide to replan its trajectory if the observed state is too far from the desired trajectory for the sake of liveness as defined in (2). Specifically, a new trajectory is planned to the goal from the observed position if the observed deviation is larger than the liveness threshold: $d(t_m) > \lambda_d$.

After replanning the trajectory: 1) the reachable sets are regenerated using the GP-based approach in Section IV-A and 2) self-triggered monitoring introduced in Section IV-B is performed until the operation is completed.

When the self-triggered monitor decides that the system needs to check its state by performing a recovery action, event-triggered replanning is again invoked to replan a trajectory for recovery. From the recovered state, an obstacle-free trajectory to the goal position is replanned again and self-triggered monitoring, recovery and replanning continue to be performed until reaching the desired goal position.

V. Simulations

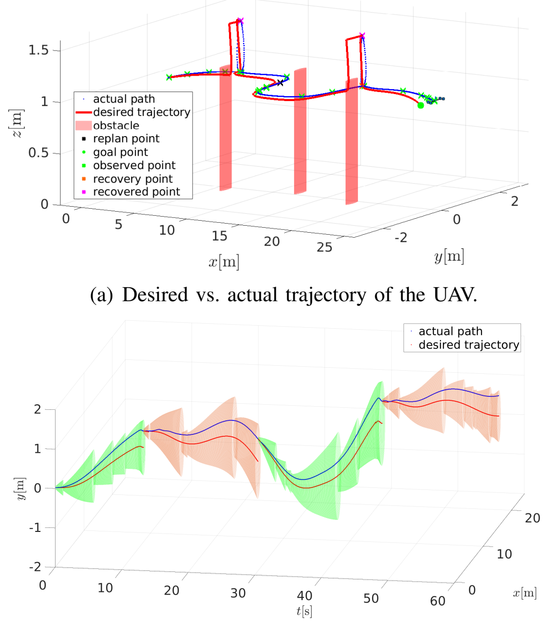
We validate the proposed run-time monitoring, recovery and replanning approach with quadrotor UAV simulations for an autonomous navigation case study in a cluttered environment. In this environment, the GPS signal is available intermittently at the altitude that the system is required to fly at, and it is always available at higher altitudes. During the simulations, we use a quadrotor UAV modeled with a 12$^{\text{th}}$ order system state and with linearized dynamics. The details of this quadrotor UAV model can be found in [16], [11].

The proposed approach is run in ten different environments with different obstacle configurations and under different wind disturbances. As a representative case, we present an environment with three cylindrical shaped obstacles as shown in Fig. 8(a). The UAV is tasked to reach a goal position $\boldsymbol{p}_g = [24, 0, 1]$ with zero velocity. The obstacles are located at $\boldsymbol{p}_{o1} = [6, -0.2], \boldsymbol{p}_{o2} = [12, 0.2], \boldsymbol{p}_{o3} = [18, -0.2]$, all of them are 1.2m tall with the radius of 0.2m. The velocity of the wind disturbance is varying over time in the y direction: $\boldsymbol{w} = [0, 0.04 + 0.01\sin(t), 0]$m/s.

In the beginning of its operation, the UAV creates an obstacle free trajectory to the goal position and computes the reachable sets based on the time to complete this trajectory using the proposed GP approach in Section IV-A.2. The reachable sets of the UAV are shown by the green and orange tubes in Fig. 8(b). Whenever a new reachable tube is generated (either for recovery, or because the deviation at monitoring time is too large) the color of the reachable tube in Fig. 8(b) swaps. At random points which are shown by green $\times$ symbols in Fig. 8(a), the sensor measurement for position becomes available and the reachable tubes are updated based on the current observed position. At these points, the reachable tubes shrink, but replanning doesn't occur (the color of the reachable tube in the figure remains the same). If the observed position is deviated from the desired trajectory more than the user defined threshold $\lambda_d = 0.5$m, the UAV replans its trajectory and recomputes the corresponding reachable sets. This replanning point is shown by a black $\times$ symbol in Fig. 8(a). At the point shown by an orange $\times$ symbol, the reachable set collides with an obstacle and the UAV triggers a recovery action to observe its state information at $z = 1.5$m level. At the point shown by a magenta $\times$ symbol, it obtains its state information and replans a trajectory accordingly.

Through the course of the entire operation, the state information of the UAV becomes available only 25 times, and the vehicle performs recovery maneuver twice to obtain its state information. A trajectory replanning due to deviation is performed once at the black $\times$ point. Even though the UAV is not able to observe its state continuously, it is able to perform its planned operation safely (i.e. no collision with an obstacle occurred) with a maximum deviation of 0.60m from its desired trajectory. Similarly, in all the other simulations, the UAV was able to finish its operation safely and the mean and standard deviation of the maximum deviation were recorded as 0.67m and 0.14m respectively.

We have used an Intel Core i7-6700HQ CPU at 2.60GHz to run these simulations and it took 0.248s on average to estimate the maximum deviation reachable set for a given trajectory, which is independent from the length of the trajectory. The simulations in order to generate the reachable sets offline for training takes 3.50s for 10 second long trajectory on an average, and this time increases/decreases linearly with the trajectory duration. Similarly, the reachable tube calculation with Ellipsoidal Toolbox increases linearly with the number of time steps used [12], and it took 2.618 seconds on an average to generate the reachable sets for a 10 second long trajectory at 40Hz. These results demonstrate that the proposed approach is able to reduce the time for reachability computation significantly toward an online implementation.



(a) Desired vs. actual trajectory of the UAV.

(b) Estimated reachable sets of the vehicle during the entire operation.

Fig. 8. UAV simulation results in an environment with three obstacles.

VI. Experiments

The proposed GP-based fast reachability and replanning approach was validated experimentally using an AscTec Hummingbird quadrotor UAV where the control commands are communicated using ROS. The ground truth state information is obtained using a Vicon motion capture system.

The training data for GP regression was collected by running a set of trajectories with different durations. During training, the quadrotor did not observe its position and the control inputs were generated as if it was following the trajectory perfectly (i.e., closing the loop with the desired states along the trajectory). Wind disturbance was created using a 24” industrial heavy duty drum fan placed on top of a mobile ground vehicle as seen in Fig. 10(a) and moved to follow the motion of the quadrotor from both sides of the room. The duration of the trajectories and the corresponding maximum deviation values were recorded and the GP regression model was built using these data following the same procedure outlined in Section IV. The resultant GP regression is displayed in Fig. 9.

Similar to the simulations, the quadrotor is tasked to visit a goal following a new previously unseen trajectory

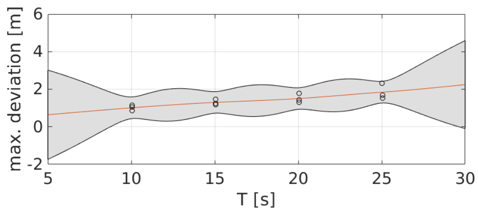
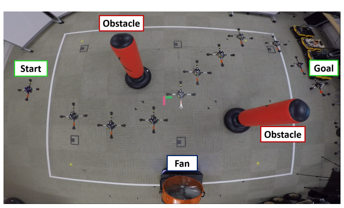


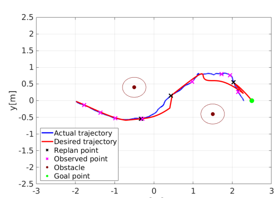
Fig. 9. GP regression of maximum deviation based on the trajectory duration.

with unknown intermittent measurements under unknown disturbances. Fig. 10(a) shows an overlapped sequence of snapshots for a waypoint navigation experiment in which the UAV is tasked to go to a goal position at $[2.5, 0, 1]$m under the wind disturbance generated by the fan at a fixed position. It should be noted that fixing the fan position generates a disturbance whose magnitude is different from, but bounded by the magnitude of the disturbance used during training. Two obstacles (inflated poles) are present along the path at the following positions: $\boldsymbol{p}_{o1} = [-0.5, 0.4]$, $\boldsymbol{p}_{o2} = [1.5, -0.4]$. In Fig. 10(b), the actual path of the quadrotor and its desired trajectory are shown by blue and red curves respectively. Similar to the simulations, at the points marked by magenta "$\times$" symbols in Fig. 10(b), the quadrotor obtains its position information and the reachable sets are shrunk. At the points shown by black "$\times$" symbols, the UAV replans its trajectory since the observed position is off more than the liveness threshold $\lambda_d = 0.4$m, thus new reachable sets are computed. Note that the actual path of the quadrotor stays inside the associated reachable sets at all the times during the experiment.

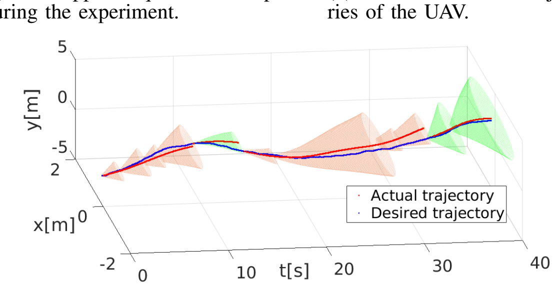
By using our approach, the quadrotor was able to complete its task without colliding with any obstacle and with a maximum deviation of 0.4109m. The position information became available only 11 times during this experiment and the quadrotor replanned its trajectory 3 times.



(a) Overlapped sequence of snapshots during the experiment.



(b) Desired vs. actual trajectories of the UAV.



(c) The reachable sets associated to the experiment in (a).

Fig. 10. Waypoint navigation experimental results.

VII. Conclusion and Future Work

In this work, we have presented a fast monitoring and replanning framework for autonomous systems under intermittent and not constant sensing and under the effect of disturbances and noises. Our approach leverages Gaussian processes theory for fast reachability. A self/event-triggered monitoring and replanning approach is also presented to guarantee safety of the system when measurements are not always available.

As a future work, we are planning to apply the proposed run-time monitoring and replanning framework to other applications focusing in particular on underwater vehicles where the problem of intermittent measurements and disturbances is particularly evident. Further, we plan to extend our work to more complicated environments with other vehicles and dynamic obstacles.

Acknowledgments

This work is sponsored by DARPA under Contract No. FA8750-18-C-0090, NSF under grant #1816591, and ONR under agreement number N000141712012. The authors thank Carmelo Di Franco for his help during the experiments.

References

[1] J. Ding, J. H. Gillula, H. Huang, M. P. Vitus, W. Zhang, and C. J. Tomlin, "Hybrid systems in robotics," *IEEE Robotics Automation Magazine*, vol. 18, no. 3, pp. 33–43, Sept 2011.

[2] V. R. Royo, D. Fridovich-Keil, S. L. Herbert, and C. J. Tomlin, "Classification-based approximate reachability with guarantees applied to safe trajectory tracking," *CoRR*, vol. abs/1803.03237, 2018. [Online]. Available: http://arxiv.org/abs/1803.03237

[3] A. P. Vinod, B. HomChaudhuri, and M. M. Oishi, "Forward stochastic reachability analysis for uncontrolled linear systems using fourier transforms," in *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*. ACM, 2017, pp. 35–44.

[4] N. Malone, K. Lesser, M. Oishi, and L. Tapia, "Stochastic reachability based motion planning for multiple moving obstacle avoidance," in *Proc. of the 17th International Conference on Hybrid Systems: Computation and Control*. New York, USA: ACM, 2014, pp. 51–60.

[5] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017.

[6] Y. Zhou, A. Raghavan, and J. S. Baras, "Time varying control set design for uav collision avoidance using reachable tubes," in *IEEE 55th Conference on Decision and Control (CDC)*, Las Vegas, NV, USA, Dec 2016, pp. 6857–6862.

[7] C. Fan, J. Kapinski, X. Jin, and S. Mitra, "Locally optimal reach set over-approximation for nonlinear systems," in *International Conference on Embedded Software*, Pittsburgh, USA, Oct 2016, pp. 1–10.

[8] R. E. Allen, A. A. Clark, J. A. Starek, and M. Pavone, "A machine learning approach for real-time reachability analysis," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Chicago, IL, USA, Sep. 2014, pp. 2202–2208.

[9] M. Liu, G. Chowdhary, B. C. da Silva, S. Liu, and J. P. How, "Gaussian processes for learning and control: A tutorial with examples," *IEEE Control Systems Magazine*, vol. 38, no. 5, pp. 53–86, Oct 2018.

[10] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 2, pp. 408–423, Feb 2015.

[11] E. Yel, T. X. Lin, and N. Bezzo, "Self-triggered adaptive planning and scheduling of UAV operations," in *IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, May 21-25*, 2018, pp. 7518–7524.

[12] A. A. Kurzhanskiy and P. Varaiya, "Ellipsoidal toolbox (et)," in *Proceedings of the 45th IEEE Conference on Decision and Control*, San Diego, CA, USA, Dec 2006, pp. 1498–1503.

[13] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE International Conference on Robotics and Automation*, Shanghai, China, May 2011, pp. 2520–2525.

[14] C. E. Rasmussen, "Gaussian processes for machine learning." MIT Press, 2006.

[15] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (gpml) toolbox," *The Journal of Machine Learning Research*, vol. 11, pp. 3011–3015, Dec. 2010.

[16] N. Bezzo, K. Mohta, C. Nowzari, I. Lee, V. Kumar, and G. Pappas, "Online planning for energy-efficient and disturbance-aware uav operations," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 5027–5033.

[17] E. Yel and N. Bezzo, "Reachability-based adaptive uav scheduling and planning in cluttered and dynamic environments," in *Workshop on Informative Path Planning and Adaptive Sampling at ICRA, Brisbane, Australia, May 21*, 2018.