Backup Plan Constrained Model Predictive Control

Hunmin Kim[†], Hyungjin Yoon[*], Wenbin Wan[†], Naira Hovakimyan[†], Lui Sha[‡], and Petros Voulgaris[*]

Abstract— This article proposes a new safety concept: backup plan safety. The backup plan safety is defined as the ability to complete one of the alternative missions in the case of primary mission abortion. To incorporate this new safety concept in control problems, we formulate a feasibility maximization problem that adopts additional (virtual) input horizons toward the alternative missions on top of the input horizon toward the primary mission. Cost functions for the primary and alternative missions construct multiple objectives, and multi-horizon inputs evaluate them. To address the feasibility maximization problem, we develop a multi-horizon multi-objective model predictive path integral control (3M) algorithm. Model predictive path integral control (MPPI) is a sampling-based scheme that can help the proposed algorithm deal with nonlinear dynamic systems and achieve computational efficiency by parallel computation. Simulations of the aerial vehicle and ground vehicle control problems demonstrate the new concept of backup plan safety and the performance of the proposed algorithm.

I. Motivation and Introduction

Traditional path planning problems for robotics and autonomous ground/aerial vehicles consider collision avoidance enough for safety. However, this consideration is not enough for emerging automated systems that perform complex tasks requiring safety criticality if we recall the incident of Miracle on the Hudson (US Airways Flight 1549) [1]. After a bird strike resulted in all engines' failure, Captain Sullenberger flew along the Hudson river, checking the feasibility of safer landing points. Additionally, the airplane ditched near boats, and this expedited rescue. This observation calls for a need for new safety definitions to cope with mission uncertainties. This paper aims to examine a novel control algorithm considering a new safety definition, which we call *backup plan safety*. The backup plan safety is defined as the ability to complete one of the alternative missions in the case of primary mission abortion. This safety definition will be particularly useful for automated systems that require a long horizon emergency response, such as aerial and nautical transportation systems; and for systems operating under mission uncertainties such as robotics, manufacturing systems, and autonomous vehicles.

A similar safety concept had been used in aircraft path planning. The Federal Aviation Administration set up the
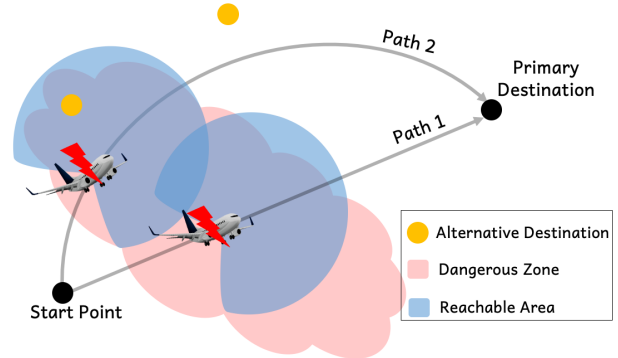

Fig. 1: (Feasibility maximization scenario) An aircraft is passing through a dangerous zone (red). The blue area indicates the reachable area after the airplane is under emergency.

60-minute rule in 1953, which allows twin-engine aircraft to fly routes no further than 60 minutes from the nearest airport suitable for an emergency landing with the aircraft's speed with one engine being inoperative. To fly outside of the 60-minute distance, one needs to follow extended-range twin-engine operational performance standards (ETOPS). Since then, the 60-min rule with ETOPS has been evaluated to enhance the safety of aircraft [2]. In particular, consider a scenario that an airplane passes through a dangerous area (e.g., a storm or a bird habitat), shown in Figure 1, toward its destination. A typical path planning solution would be finding the shortest path (Path 1). However, following this path, the airplane has no viable means to safely land at an airport when it confronts an emergency, possibly with limited performance. Path 2 tries to maximize the feasibility of the safe landing given two alternative destinations. Even when an emergency takes place, one of the two alternative destinations is still feasible for the airplane. The backup plan safety proposed in the current paper is a generalization of such rule in terms of the safety standard and application domains.

Model predictive control (MPC) is a general iterative optimal control methodology for a finite control horizon, satisfying a set of constraints. MPC has shown its superiority in the path and motion planning domain for stability and safety [3], [4]. MPC can be implemented for real-time use due to the advancement of computing hardware and algorithmic developments. Recent years have seen efforts regarding integrating machine learning with MPC methods towards establishing a unified framework for learning-based planning and control for enhanced safe autonomy [5], [6]. Despite those algorithmic developments, the safety destination in literature is yet limited to collision avoidance. The devel-

[†]Hunmin Kim, Wenbin Wan, and Naira Hovakimyan are with the Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, USA. {hunmin, wenbinw2, nhovakim}@illinois.edu

[‡]Lui Sha is with the Department of Computer Science, University of Illinois at Urbana-Champaign, USA. lrs@illinois.edu

[*]Hyungjin Yoon and Petros Voulgaris are with the Department of Mechanical Engineering, University of Nevada, Reno, USA. {hyungjiny, pvoulgaris}@unr.edu

opment of multi-objective MPC (MMPC) [7] could provide a basis for multi-mission control problems for backup plan safety, where each cost function is associated with a mission. MMPC shows its effectiveness in various applications such as power converter control [8], [9], HVAC [10], and cruise control [11], [12]. However, a single prediction horizon used in MPC and MMPC is not enough to address backup plan safety, because cost functions for alternative missions are well evaluated only when the trajectories toward the corresponding mission are given (e.g., mission feasibility).

Stochastic model predictive control (SMPC) exploits the probabilistic uncertainty model in an optimal control problem formulation. SMPC balances the tradeoff between optimizing control objectives and satisfying chance constraints. Typical strategies to address SMPC include stochastic-tube [13], [14], stochastic programming [15], and sampling-based approaches [16]–[20]. The current paper focuses on a sampling-based approach to handle nonlinear dynamic systems and enable efficient parallel computing using Graphics Processing Units (GPUs). In particular, our algorithm is based on model predictive path integral control (MPPI) [18]–[20] that relies on a generalized importance sampling scheme.

Contribution. The current article proposes a new safety concept - backup plan safety, to enhance the operation of complex autonomous systems under mission uncertainties. The backup plan constrained control problem is formulated as a feasibility maximization problem: MMPC with multi-horizon inputs. We develop the multi-objective multi-horizon model predictive path integral control (3M) algorithm to address the feasibility maximization problem. In particular, MMPC with multi-horizon inputs can handle the multi-mission problem, in which multi-horizon inputs help evaluate all cost functions. By enabling parallel computation using GPUs, MPPI control expedites the computing speed of complex optimization problems with multi-horizon inputs. MPPI control further helps to avoid harmful computation delays by finishing computation in a designated time. Simulations of aerial vehicle and ground vehicle control problems present the new safety concept and the performance of the proposed algorithm.

The remainder of the paper is organized as follows. Section II formulates the feasibility maximization problem with multi-horizon inputs toward the primary and alternative missions. Section III-B introduces MPPI control and proposes the 3M algorithm to address the feasibility maximization problem. Simulation results of an aerial vehicle and a ground vehicle are presented in Section IV.

II. Feasibility Maximization

Consider the discrete-time switched dynamic systems:

$$x_{k+1} = f(x_k, u_k, j), \tag{1}$$

where $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{N} \to \mathbb{R}^{n_x}$ is a nonlinear system function, $x_k \in \mathbb{R}^{n_x}$ is the system state, $u_k \in \mathbb{R}^{n_u}$ is the control input at time $k \geq 0$. Mode index $j \in \mathcal{J} \subset \mathbb{N}$ determines the function f, where $\mathcal{J}$ is the mode index set.

Remark 2.1: The switched system model in (1) can represent changing dynamic models depending on time and events during the operation. For instance, in the aircraft control problem in Figure 1, the dynamic system model moving toward alternative destinations can be a single-engine failure model. ∎

We assume that the control authority knows one primary mission and m alternative missions. It is said that the mission i is completed at time t if

$$\begin{aligned} t &= \arg\min_k k \\ &\text{s.t. } d(x_k, p^i) = 0, \end{aligned} \tag{2}$$

where $d : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{R}_{\geq 0}$ is the distance metric between the internal system state x_k and mission state $p^i \in \mathbb{R}^{n_x}$. The control objective for the system (1) is to complete the primary mission, and if the primary mission is aborted in the middle of the operation, then the system should complete one of the alternative missions instead.

An alternative mission may not align with the primary mission. Therefore, the control problem should balance between the primary and alternative cost functions. Considering this fact, we formulate the control problem of the system (1) as a feasibility maximization:

$$\begin{aligned} \min_{\mathbf{U}} &\ \mathbf{J}(\mathbf{X}, \mathbf{U}) \\ s.t. &\ x_{k+1} = f(x_k, u_k, j), \end{aligned} \tag{3}$$

where $\mathbf{X} \in \mathbb{R}^{(N+1+\frac{N(N-1)m}{2})n_x}$ is the collection of states, $\mathbf{U} \in \mathbb{R}^{(N+\frac{N(N-1)m}{2})n_u}$ is the collection of inputs, and $\mathbf{J} : \mathbb{R}^{(N+1+\frac{N(N-1)m}{2})n_x} \times \mathbb{R}^{(N+\frac{N(N-1)m}{2})n_u} \to \mathbb{R}^{m+1}$ is an $m+1$ dimensional vector cost function. Each cost function is associated with a primary or alternative mission. The problem contains finite $N \in \mathbb{N}$ prediction horizon control inputs at each time step t (i.e., $[t, t+N-1]$) that minimize the pre-designed multiple cost functions. After executing the first control input, the prediction horizon will be shifted forward and optimize the cost function again as in the standard MPC. We assume that the cost function $\mathbf{J}$ includes soft constraints. In particular, one could use the Lagrange multiplier method to realize state/input constraints and feasibility guarantee.

The key difference of the feasibility maximization problem (3) from the multi-objective MPC in [7] is that the input $\mathbf{U}$ in (3) includes an input horizon toward the primary mission and additional (virtual) input horizons toward alternative missions. The multi-horizon control inputs help to evaluate the cost function toward both the primary and alternative missions. Section II-A discusses the multi-horizon control inputs $\mathbf{U}$, state trajectories $\mathbf{X}$, and their dimensions. Section II-B discusses the cost function $\mathbf{J}$.

The feasibility maximization problem in (3) is a family of multi-objective optimization problems, where the elements of cost functions $\mathbf{J}(\mathbf{X}, \mathbf{U})$ are conflicting with each other in general, and there is no solution that minimizes all the cost functions simultaneously. As in multi-objective optimization [7], [21], Pareto optimality plays a key role in defining optimality.

Definition 2.1 (Pareto optimality [7]): A feasible solution is Pareto optimal if and only if it is not dominated by any other feasible solution. In other words, feasible solution $\mathbf{V}$ is Pareto optimal if and only if there is no feasible $\mathbf{U}$ such that $\mathbf{J}(\mathbf{X}, \mathbf{U}) \leq \mathbf{J}(\mathbf{X}, \mathbf{V})$ holds element-wise, and strict inequality holds for at least one element.

Multi-objective genetic algorithm [22] is particularly useful to identify Pareto optimal sets and corresponding Pareto frontier, the set of Pareto optimal cost function values. However, they may not be used in real-time applications due to the computational complexity. In this case, we can assign parameterized weights to the cost functions so that we limit our focus to the particular subset of solutions. With the weight assignment, the feasibility maximization problem (3) can be reformulated by

$$\min_{\mathbf{U}} \alpha^\top \mathbf{J}(\mathbf{X}, \mathbf{U})$$
$$s.t.\ x_{k+1} = f(x_k, u_k, j), \tag{4}$$

where the weight vector $\alpha = [\alpha^0, \cdots, \alpha^m]^\top \in \mathbb{R}^{m+1}$ satisfies $\alpha^i \in [0, 1] \subset \mathbb{R}$ and $\sum_{i=0}^m \alpha^i = 1$. The solution of (4) is a Pareto optimal solution of (3) if $\alpha^i > 0$ for $\forall i$ [23], while this may not hold when there exist some indices such that $\alpha^i = 0$. The choice of α governs the valuation on each mission, resulting in different optimal control sequences. Section II-C discusses how to choose the weight vector α.

A. Multi-horizon Inputs and Trajectories

The control input

$$\mathbf{U} = [(\mathbf{U}^0)^\top, (\mathbf{U}^1)^\top, \cdots, (\mathbf{U}^m)^\top]^\top \tag{5}$$

consists of inputs toward the primary mission

$$\mathbf{U}^0 = [u_0^\top, u_1^\top, \cdots, u_{N-2}^\top, u_{N-1}^\top]^\top$$

and additional (virtual) input horizons toward the alternative missions $\mathbf{U}^i$ for $i = 1, \cdots, m$, where $\mathbf{U}^0$ is the input horizon used in the standard MPC.

Two important properties should be considered when defining the inputs toward the alternative missions $\mathbf{U}^i$:

- It is unknown when the system will abort the mission;
- The control horizon toward the alternative missions should be N.

It is required for $\mathbf{U}^i$ to consider the possibility of mission abortion at every point for the first property. Accordingly, We construct $\mathbf{U}^i$ as
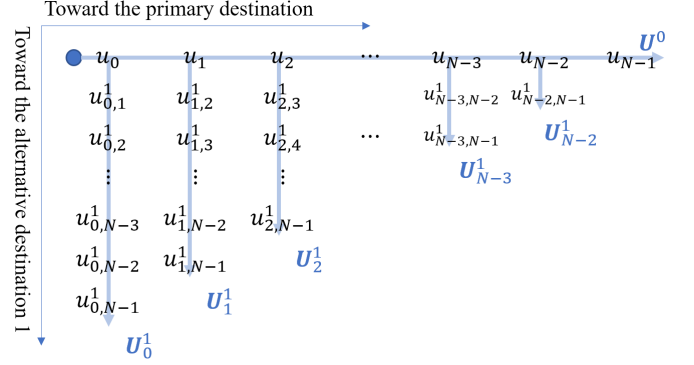
$$\mathbf{U}^i = [(\mathbf{U}_0^i)^\top, (\mathbf{U}_1^i)^\top, \cdots, (\mathbf{U}_{N-3}^i)^\top, (\mathbf{U}_{N-2}^i)^\top]^\top, \tag{6}$$

where $\mathbf{U}_p^i$ is the control sequence toward the ith alternative mission when the primary mission is aborted after u_p has been executed. By this definition, the first $p+1$ control inputs of $\mathbf{U}_p^i$ are $u_0, \cdots, u_p$. Because the dimension of $\mathbf{U}_p^i$ should be $\mathbf{U}_p^i \in \mathbb{R}^{N \times n_u}$ to satisfy the second property, there are additional $N-(p+1)$ number of inputs toward the alternative mission i, in $\mathbf{U}_p^i$. Then, we have
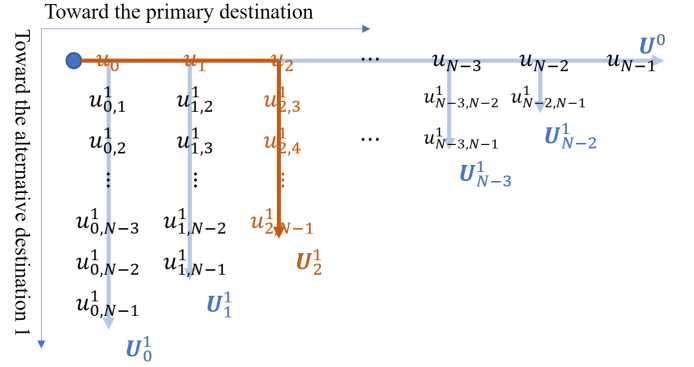
$$\mathbf{U}_p^i = [u_0^\top, \cdots, u_p^\top, (u_{p,p+1}^i)^\top, \cdots, (u_{p,N-1}^i)^\top]^\top,$$

where $u_{p,q}^i$ is the qth control input when we decide to abort the primary mission after executing u_p and choose i^{th} alternative mission for the next.

Input $\mathbf{U}$ is visualized in Figure 2 when $m = 1$, which presents the relation between $\mathbf{U}$, $\mathbf{U}^0$, and $\mathbf{U}_p^1$, and shows how u_p, and $u_{p,q}^i$ can construct $\mathbf{U}^0$ and $\mathbf{U}_p^1$.

(a) The elements of input $\mathbf{U}$ are presented in an array. To construct $\mathbf{U}^0$ or $\mathbf{U}_p^1$, one can start at the blue dot at the left upper corner, and move right until the primary mission is aborted, and move downside after the primary mission is aborted and the alternative mission 1 is chosen for the next mission.

(b) An example of the way to construct $\mathbf{U}_2^1 = [u_0^\top, u_1^\top, u_2^\top, (u_{2,3}^1)^\top, (u_{2,4}^1)^\top, \cdots, (u_{2,N-1}^1)^\top]^\top$ (marked in red). The first three elements are the elements of $\mathbf{U}^0$.

Fig. 2: Visualization of inputs $\mathbf{U}$ when $m = 1$.

The states $\mathbf{X} = [(\mathbf{X}^0)^\top, (\mathbf{X}^1)^\top, \cdots, (\mathbf{X}^m)^\top]^\top$ are constructed by simulating the input $\mathbf{U}$ on the system (1), where $\mathbf{X}^0$ and $\mathbf{X}^i = [(\mathbf{X}_0^i)^\top, \cdots, (\mathbf{X}_{N-2}^i)^\top]^\top$ are the simulated states of inputs $\mathbf{U}^0$ and $\mathbf{U}^i$, respectively for $i = 1, \cdots, m$. Likewise, $\mathbf{X}_p^i = [x_0^\top, \cdots, x_p^\top, (x_{p,p+1}^i)^\top, \cdots, (x_{p,N}^i)^\top]^\top$ is the simulated state of the input $\mathbf{U}_p^i$, for $i = 1, \cdots, m$ and $p = 0, \cdots, N-2$.

Remark 2.2 (Dimension of states and inputs): It is worth re-emphasizing that the first $p + 1$ elements of $\mathbf{U}_p^i$ are those of $\mathbf{U}^0$. Correspondingly, the first $p + 2$ elements of $\mathbf{X}_p^i$ are $x_0, x_1, \cdots, x_p, x_{p+1}$, which are the elements of $\mathbf{X}^0$. Therefore, the input $\mathbf{U}$ consists of the N number of inputs toward the primary mission in $\mathbf{U}^0$, and $N-1, N-2, \cdots, 1$ additional number of inputs toward the alternative mission i in $\mathbf{U}_0^i, \mathbf{U}_1^i, \cdots, \mathbf{U}_{N-2}^i$. Therefore, $\mathbf{U}$ consists of $N + \frac{N(N-1)}{2}m$ independent elements, and $\mathbf{X}$ consists of $N + 1 + \frac{N(N-1)}{2}m$ independent elements.

The dimensions of independent input and state variables are large compared to the standard MPC. We will deal with the induced computational complexity by using MPPI control described in Section III-A. ∎

B. Multi-objective Cost Functions

The cost function

$$\mathbf{J}(\mathbf{X}, \mathbf{U}) \triangleq [\mathbf{J}^0(\mathbf{X}^0, \mathbf{U}^0), \mathbf{J}^1(\mathbf{X}^1, \mathbf{U}^1), \cdots, \mathbf{J}^m(\mathbf{X}^m, \mathbf{U}^m)]^\top$$

is an $m+1$ dimensional vector function, where the elements are the average cost over state-input trajectories toward the corresponding mission:

$$\mathbf{J}^0(\mathbf{X}^0, \mathbf{U}^0) = J^0(\mathbf{X}^0, \mathbf{U}^0)$$

$$\mathbf{J}^i(\mathbf{X}^i, \mathbf{U}^i) = \frac{1}{N-1} \sum_{p=0}^{N-2} J^i(\mathbf{X}_p^i, \mathbf{U}_p^i)$$

for $i = 1, \cdots, m$. The function J^i is a standard cost function for the mission i:

$$J^i(\mathbf{X}^i, \mathbf{U}^i) = \sum_{k=1}^{N} L^i(\mathbf{X}^i(k), \mathbf{U}^i(k)) + F^i(\mathbf{X}^i(N+1))$$

(7)

that consists of the cost-to-go L^i and the terminal cost F^i, where $\mathbf{X}^i(k) \in \mathbb{R}^{n_x}$ and $\mathbf{U}^i(k) \in \mathbb{R}^{n_u}$ are the kth state and input of $\mathbf{X}^i$ and $\mathbf{U}^i$, respectively.

C. Weight Vector

When choosing the weight vector α in the feasibility maximization (4), we should consider two important issues. First of all, for any fixed α (except for $\alpha = [1, 0, \cdots, 0]^\top$), there may not exist Pareto optimal that achieves the primary mission because the feasibility maximization problem (4) tries to minimize the weighted sum of the cost functions of the primary and alternative missions. Second, the choice of α will affect the closed-loop stability and performance. In what follows, we design the desired weight vector α_d to address the first issue such that it converges to $\alpha = [1, 0, \cdots, 0]^\top$ as the system is about to achieve the primary mission. Furthermore, we will adopt the optimization-based weight update law for closed-loop stability.

The desired weight vector $\alpha_d(x_t)$ is designed as follows:

$$\alpha_d(x_t) = [1 - \gamma + w^0(x_t)\gamma, w^1(x_t)\gamma, \cdots, w^m(x_t)\gamma]^\top,$$

(8)

where

$$w^i = \frac{exp(-\frac{1}{\lambda_\alpha}d(x_t, p^i))}{\sum_{l=0}^{m} exp(-\frac{1}{\lambda_\alpha}d(x_t, p^l))}$$

is the Gibbs distribution with temperature parameter λ_α, and $0 < 1 - \gamma \leq 1$ is the pre-determined weight on the primary mission. Function d is the distance metric between the state and mission state. It can be verified that α_d in (8) satisfies the constraints on α (i.e., $\alpha_d^\top \mathbf{1} = 1$ and $\alpha_d \in [0, 1] \subset \mathbb{R}$), and converges to $[1, 0, \cdots, 0]^\top$ as $d(x_t, p_0)$ decreases to zero.

Let us define α_t and α_{t-1} as α in (4) chosen at the current time t and the one at the previous time $t - 1$, respectively.

Now, α_t must be chosen close to its desired value $\alpha_d(x_t)$, while its choice guarantees closed-loop stability as in [7]:

$$\alpha_t = \arg\min_\alpha h(\alpha - \alpha_d(x_t))$$
$$\text{s.t. } \alpha^\top(\mathbf{J}(\hat{\mathbf{X}}_{t-1}, \hat{\mathbf{U}}_{t-1}) - \mathbf{F}(\hat{\mathbf{X}}_{t-1}, \hat{\mathbf{U}}_{t-1}))$$
$$\leq \alpha_{t-1}^\top(\mathbf{J}(\hat{\mathbf{X}}_{t-1}, \hat{\mathbf{U}}_{t-1}) - \mathbf{F}(\hat{\mathbf{X}}_{t-1}, \hat{\mathbf{U}}_{t-1})),$$
$$\alpha^\top \mathbf{1} = 1,$$
$$\alpha_i \geq 0, \ i = 0, \cdots, m,$$

(9)

where h is a convex cost function that penalizes the difference between α_t and $\alpha_d(x_t)$.

As in (5) and (6), the input $\hat{\mathbf{U}}_{t-1}$ is defined by $\hat{\mathbf{U}}_{t-1} = [(\hat{\mathbf{U}}_{t-1}^0)^\top, (\hat{\mathbf{U}}_{t-1}^1)^\top, \cdots, (\hat{\mathbf{U}}_{t-1}^m)^\top]^\top$, where $\hat{\mathbf{U}}_{t-1}^i = [(\hat{\mathbf{U}}_{0,t-1}^i)^\top, \cdots, (\hat{\mathbf{U}}_{N-2,t-1}^i)^\top]^\top$ for $i = 1, \cdots, m$. The input $\hat{\mathbf{U}}_{t-1}$ is constructed from the optimal control input at the previous step $\mathbf{U}_{t-1}^i$. In particular, we have $\hat{\mathbf{U}}_{t-1}^0$ and $\hat{\mathbf{U}}_{p,t-1}^i$ by removing the first input u_0 (which has been already executed) and appending a zero vector $\mathbf{0}_{n_u} \in \mathbb{R}^{n_u}$ at the end to $\mathbf{U}_{t-1}^0$ and $\mathbf{U}_{p,t-1}^i$, i.e.,

$$\hat{\mathbf{U}}_{t-1}^0 = [u_1^\top, \cdots, u_{N-1}^\top, \mathbf{0}_{n_u}^\top]^\top$$
$$\hat{\mathbf{U}}_{p,t-1}^i = [u_1^\top, \cdots, u_p^\top, (u_{p,p+1}^i)^\top, \cdots, (u_{p,N-1}^i)^\top, \mathbf{0}_{n_u}^\top]^\top$$

(10)

for $i = 1, \cdots, m$ and $p = 1, \cdots, N - 2$. The state $\hat{\mathbf{X}}_{t-1}$ is the corresponding simulated trajectory of the input $\hat{\mathbf{U}}_{t-1}^i$. The vector function

$$\mathbf{F}(\hat{\mathbf{X}}_{t-1}, \hat{\mathbf{U}}_{t-1})$$
$$= [L^0(\hat{\mathbf{X}}_{t-1}^0(N), \hat{\mathbf{U}}_{t-1}^0(N)) + F^0(\hat{\mathbf{X}}_{t-1}^0(N+1)),$$
$$\sum_{p=0}^{N-2} \frac{L^1(\hat{\mathbf{X}}_{p,t-1}^1(N), \hat{\mathbf{U}}_{p,t-1}^1(N)) + F^1(\hat{\mathbf{X}}_{p,t-1}^1(N+1))}{N-1},$$
$$\cdots,$$
$$\sum_{p=0}^{N-2} \frac{L^m(\hat{\mathbf{X}}_{p,t-1}^m(N), \hat{\mathbf{U}}_{p,t-1}^m(N)) + F^m(\hat{\mathbf{X}}_{p,t-1}^m(N+1))}{N-1}]^\top$$

is the last cost-to-go function and terminal cost. Now that the last control input in $\hat{\mathbf{U}}_{t-1}$ is a dummy $\mathbf{0}_{n_u}$, we should not take into account the cost incurred by the dummy. The first constraint in (9) implies that α_t should be chosen such that the value function $\alpha^\top(\mathbf{J}(\hat{\mathbf{X}}_{t-1}, \hat{\mathbf{U}}_{t-1}) - \mathbf{F}(\hat{\mathbf{X}}_{t-1}, \hat{\mathbf{U}}_{t-1}))$ is decreasing for the closed-loop stability. If the cost function h is quadratic, then the problem (9) becomes a quadratic programming problem because all the constraints are linear with respect to the decision variable α.

III. MULTI-OBJECTIVE MULTI-HORIZON MODEL PREDICTIVE PATH INTEGRAL CONTROL (3M)

The current section proposes a 3M algorithm to address the feasibility maximization problem (4). Given that the system model (1) is nonlinear, and the dimension of input $\mathbf{U}$ is large, the proposed algorithm is based on MPPI control that is a sampling-based and parallel computable MPC. Section III-A introduces MPPI control, and Section III-B presents the 3M algorithm.

A. Model Predictive Path Integral Control (MPPI)

The MPPI control algorithm solves stochastic optimal control problems based on the (stochastic) sampling of the system trajectories through parallel computation [18]–[20]. Due to the sampling nature, the algorithm does not require derivatives of either the dynamics or the cost function of the system, which enables to handle nonlinear dynamics and non-smooth/non-differentiable cost functions without approximations. With the help of GPUs for expediting the parallel computation, the MPPI can be implemented in real-time even for relatively large dimensions of the state space (e.g., there are 48 state variables for the 3-quadrotor control example in [19]). The computational efficiency from paralleled stochastic sampling and the ability to directly handle non-smooth cost functions make MPPI appealing for real-time control problems.

Consider the dynamic system (1) with a fixed j (and thus omitted) and a noise corrupted input:

$$x_{k+1} = f(x_k, u_k + \epsilon_k),$$

where $\epsilon_k \in \mathbb{R}^{n_u}$ is an independent and identically distributed (i.i.d.) Gaussian noise, i.e., $\epsilon_k \sim \mathcal{N}(0, \Sigma)$ with the known co-variance matrix Σ. Given a finite time horizon N, the goal of the optimization problem is to find an input trajectory $\mathbf{U}^0 = [u_0^\top, u_1^\top, \cdots, u_{N-1}^\top]^\top$ that minimizes the expected cost over all trajectories:

$$\mathbf{U}^* = \arg\min_{\mathbf{U}^0} \mathbb{E}[S(\tau)],$$

where $\tau = \{x_0, u_0, x_1, u_1, \cdots, u_{N-1}, x_N\}$. The cost function of a trajectory is given as follows:

$$S(\tau) = \phi(x_N) + \sum_{t=0}^{N-1} \left(c(x_t) + \lambda u_t^\top \Sigma^{-1} \epsilon_t \right), \quad (11)$$

where $\phi(x_N)$ is the terminal cost, $c(x_t)$ is the state-dependent running cost, and λ is a parameter discussed later. The cost function (11) depends on unknown random variable ϵ_k for $k = 0, \cdots, N-1$. MPPI control relies on a sampling-based method to evaluate the cost (11). In particular, we sample ϵ_k from the distribution $\mathcal{N}(0, \Sigma)$, and construct K trajectories of noises $\epsilon^q = [(\epsilon_0^q)^\top, (\epsilon_1^q)^\top, \cdots, (\epsilon_{N-1}^q)^\top]$ for $q = 1, \cdots, K$. The cost function S can be evaluated for each trajectory ϵ^q. Furthermore, MPPI control adopts the iterative update law [19] to obtain the current optimal input $\mathbf{U}^0$ around the previous optimal input $\mathbf{U}_{t-1}^0$ as follows:

$$\mathbf{U}^0 = [u_{1,t-1}^\top, u_{2,t-1}^\top, \cdots, u_{N-1,t-1}^\top, \mathbf{0}^\top]^\top + \sum_{q=1}^K w^q \epsilon^q,$$

where

$$w^q = \frac{exp(-\frac{1}{\lambda} S(\tau^q))}{\sum_{q=1}^K exp(-\frac{1}{\lambda} S(\tau^q))}$$

with λ known as the temperature parameter of the Gibbs distribution (or Softmax function), and τ^q as state-input sets for q^{th} noise trajectory. Input $u_{i,t-1}$ is the ith input of the previous optimal input $\mathbf{U}_{t-1}^0$.

Algorithm 1 3M algorithm

Choose tuning parameters:
K: Number of sample trajectories;
N: The size of control horizon;
m: The number of alternative missions;
Σ: Co-variance of the noise ϵ_k;
λ_α, λ: Temperature parameter of the Gibbs distribution;
γ: Weight on alternative missions;
$\mathbf{U}_0$: Initial input sequence;

1: Measure current state x_t and get $\alpha_d(x_t)$;
2: Obtain $\hat{\mathbf{U}}_{t-1}^0$ and $\hat{\mathbf{U}}_{p,t-1}^i$ in (10) for $i = 1, \cdots, m$ and $p = 1, \cdots, N-2$ from $\mathbf{U}_{t-1}$;
3: Update α_t by (9);
4: Sample K trajectories of noise ϵ^q as in (12);
5: Simulate $\hat{\mathbf{U}}_{t-1} + \epsilon^q$ on the system (1) to get $\hat{\mathbf{X}}_{t-1}^q$ for $q = 1, \cdots, K$;
6: Evaluate $\alpha_t^\top \mathbf{J}^q$ for $q = 1, \cdots, K$, where $\mathbf{J}^q = \mathbf{J}(\hat{\mathbf{X}}_{t-1}^q, \hat{\mathbf{U}}_{t-1} + \epsilon^q)$;
7: Calculate estimated optimal control $\mathbf{U}_t$ in (13) using the costs of the K trajectories, $(\alpha_t^\top \mathbf{J}^1, \cdots, \alpha_t^\top \mathbf{J}^K)$.

B. Multi-objective Multi-horizon Model Predictive Path Integral Control (3M)

Applying MPPI control in Section III-A to the feasibility maximization problem in Section II, we proposed the multi-objective multi-horizon model predictive path integral control (3M) algorithm. The proposed 3M algorithm is summarized in Algorithm 1, and explained below.

Given the current state x_t, the desired weight vector α_d can be calculated by (8) (line 1). Given the desired weight vector α_d, the α in (4) can be selected close to the desired value α_d as in (9) (line 3), where the input $\hat{\mathbf{U}}_{t-1}$ is constructed from the previous input $\mathbf{U}_{t-1}$ (line 2).

We sample K trajectories of noises as follows (line 4):

$$\epsilon^q \in \mathbb{R}^{(N + \frac{N(N-1)}{2}m)n_u}, \quad \epsilon^q(i) \sim \mathcal{N}(0, \Sigma) \quad (12)$$

for $q = 1, \cdots, K$ and $i = 1, \cdots, N + \frac{N(N-1)}{2}m$, where $\epsilon^q(i) \in \mathbb{R}^{n_u}$ is the ith noise vector of ϵ^q.

For each sampled noise trajectory, we construct noise disturbed input $\hat{\mathbf{U}}_{t-1} + \epsilon^q$, and evaluate the corresponding cost as (line 6):

$$\alpha_t \mathbf{J}^q = \alpha_t \mathbf{J}(\hat{\mathbf{X}}_{t-1}^q, \hat{\mathbf{U}}_{t-1} + \epsilon^q) \quad \text{for } q = 1, \cdots, K,$$

where $\hat{\mathbf{X}}_{t-1}^q$ is the simulated state trajectory with the noise disturbed input $\hat{\mathbf{U}}_{t-1} + \epsilon^q$ on the system (1) (line 5). The optimal control input is estimated by the weight-average of the costs calculated from the K trajectories, $(\alpha_t^\top \mathbf{J}^1, \cdots, \alpha_t^\top \mathbf{J}^K)$, as (line 7):

$$\mathbf{U}_t = \hat{\mathbf{U}}_{t-1} + \sum_{q=1}^K \boldsymbol{w}^q \epsilon^q, \quad (13)$$

where

$$\boldsymbol{w}^q = \frac{exp(-\frac{1}{\lambda} \alpha_t^\top \mathbf{J}^q)}{\sum_{q=1}^K exp(-\frac{1}{\lambda} \alpha_t^\top \mathbf{J}^q)} \in \mathbb{R}_{>0}, \quad \text{for } q = 1, \cdots, K.$$

If $\gamma = 0$, then the 3M algorithm reduces to the MPPI control. This is because the input toward the primary mission $\mathbf{U}^0$ optimizes the primary cost function $\mathbf{J}^0$ only, and the inputs toward the alternative missions $\mathbf{U}^i$ do not affect the optimization problem. Furthermore, $\alpha = [1, 0, \cdots, 0]^\top$ is the unique optimal solution of the problem (9) and is recursively feasible if the initial condition is $\alpha_0 = [1, 0, \cdots, 0]^\top$.

IV. ILLUSTRATIVE EXAMPLES

Simulations have been conducted for the motion planning problem of the unmanned aerial vehicle (UAV) and unmanned ground vehicle (UGV) with and without obstacles. The control objective of the simulations is to move the vehicle from the initial position $[0, 0]$ to the primary destination $[10, 10]$ in the 2-D plane. The UAV simulations represent an urban drone delivery system, where the drone flies 200-500 ft above ground and delivers a package to the primary destination. The alternative destinations can be seen as an emergency landing spot such as a safe rooftop. The UGV simulations adopt a more realistic dynamic system, where alternative destinations represent a place that can accommodate car repair and fueling/charging services. We use GPUs for parallel computation and discuss control frequency.

A. Simulations on UAV

We consider a double integrator model to simulate the UAV control problem:

$$x_{k+1} = \begin{bmatrix} 1 & 0 & 0.1 & 0 \\ 0 & 1 & 0 & 0.1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} u_k,$$

where $x_k \in \mathbb{R}^4$ represents the horizontal coordinate, vertical coordinate, horizontal velocity, and vertical velocity. The input $u_k \in \mathbb{R}^2$ consists of horizontal and vertical accelerations. The initial condition is $x_0 = [0, 0, 0, 0]^\top$ and the primary destination is $p^0 = [10, 10, 0, 0]^\top$. Euclidean distance has been used for the distance metric d in (2) and (8). The cost function in (7) is constructed as $L^i(x, u) = (x - p^i)^\top Q(x - p^i) + u^\top R u$, $F^i(x) = (x - p^i)^\top Q(x - p^i)$ with $Q = \mathbb{I}$ and $R = \mathbb{I}$, where $\mathbb{I}$ is the identity matrix with a proper dimension. We use the desired weight vector α_d in (8) with $\lambda_\alpha = 1$. The cost function h in the optimization problem (9) is chosen as a quadratic function $h(\alpha - \alpha_d(x_k)) = (\alpha - \alpha_d(x_k))^\top(\alpha - \alpha_d(x_k))$. The MPPI control parameters are $\Sigma = \mathbb{I}$ and $\lambda = 0.5$. The weight γ in (8), control horizon N, and the number of sample trajectories K are chosen differently for each simulation, and those parameters are presented in each figure. We compare the backup plan safety constrained control with the MPPI control ($\gamma = 0$).

1) Obstacle-free environment: Two simulations have been conducted with two different alternative destinations ($m = 2$); for the first simulation, mission states are given by $p^1 = [2, 6, 0, 0]^\top$ and $p^2 = [8, 6, 0, 0]^\top$; for the second simulation, mission states are $p^1 = [2, 8, 0, 0]^\top$ and $p^2 = [6, 12, 0, 0]^\top$.

The results are presented in Figures 3 and 4, where the primary destination is marked in a blue dot, and alternative
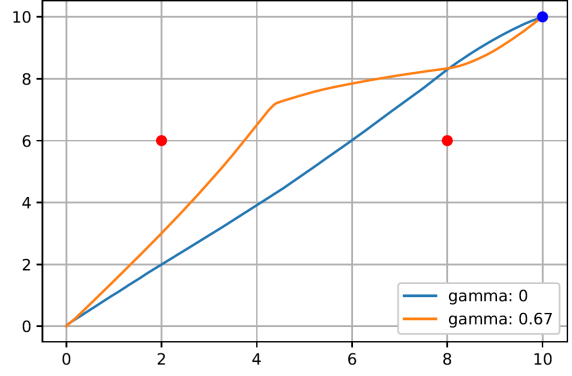

Fig. 3: UAV simulation in an obstacle-free environment with $N = 10$ and $K = 1000$. The orange line is the executed state trajectory of the UAV for backup plan constrained control, and the blue line is the trajectory for the regular MPPI toward the primary destination.
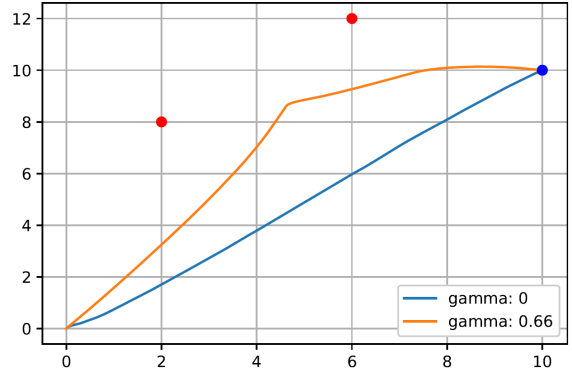

Fig. 4: UAV simulation in an obstacle-free environment with $N = 10$ and $K = 1000$.

destinations are marked in red dots. The UAV with MPPI control ($\gamma = 0$) flies to the primary destination directly as expected. The UAV with the 3M algorithm makes a detour to the primary destination in both cases, flying near alternative destinations. The detour trajectory is safer in the backup plan sense, providing a shorter path toward one of the alternative destinations when an emergency landing is in need.

It is essential to choose a set of suitable alternative missions to be partially aligned with the primary mission. If not, the proposed 3M algorithm automatically less considers conflicting alternative missions over time by adopting a distance-based update law for the desired weight vector. For example, if the alternative destinations are located in the opposite direction to the primary destination ($p^1 = [-4, -4, 0, 0]^\top$ and $p^2 = [-4, 8, 0, 0]^\top$), the trajectory with $\gamma = 0.66$ has a subtle difference from that with $\gamma = 0$ as shown in Figure 5.

2) Environment with obstacle: Two alternative destinations are located at $p^1 = [2, 6, 0, 0]^\top$ and $p^2 = [6, 12, 0, 0]^\top$. A soft constraint renders the collision avoidance constraint.

Figure 6 shows the simulation results, where the blue boxes are obstacles. The UAV flies near the safe rooftops, avoiding obstacles instead of choosing the shortest path. When $\gamma = 0$, the UAV arrives at the destination with a hook shape trajectory to land with zero velocity.
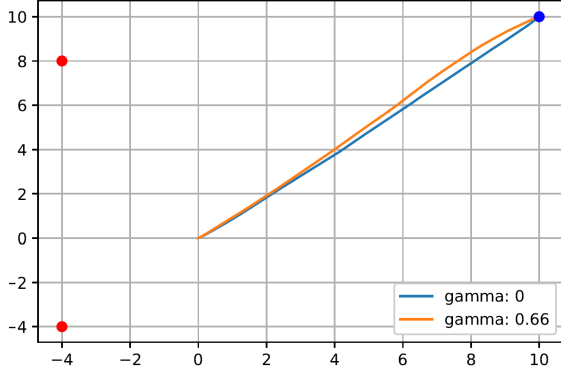
Fig. 5: UAV simulation in an obstacle-free environment with $N = 10$ and $K = 1000$. The alternative destinations are located in the opposite direction to the primary destination.
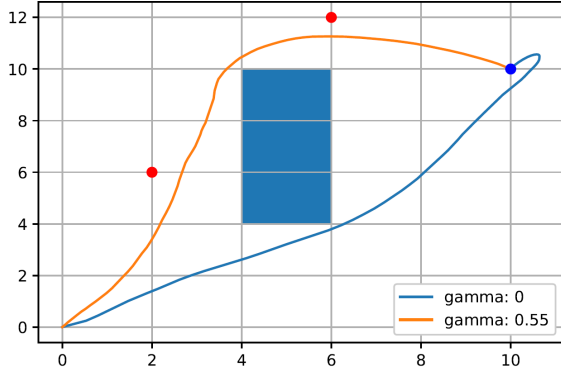

Fig. 6: UAV simulation in an environment with obstacles with $N = 20$ and $K = 1000$.

B. Simulations on UGV

The UGV simulations use the simple car dynamic model described by (Chapter 13.1.2 in [24]):

$$p_{k+1}^x = p_k^x + (v_t \cos \theta_k)\delta$$
$$p_{k+1}^y = p_k^y + (v_t \sin \theta_k)\delta$$
$$\theta_{k+1} = \theta_k + (\frac{v_t}{L} \tan \phi_k)\delta,$$

where $x_k = [p_k^x, p_k^y, \theta_k]^\top \in \mathbb{R}^3$ represents horizontal coordinate, vertical coordinate, and heading angle, respectively. Input $u_k = [v_k, \phi_k]^\top$ consists of velocity and steering angle. Parameter $L = 0.2$ is a wheelbase, and $\delta = 0.1$ is the time step. Other functions and parameters remain unchanged, if not specified. There are two alternative destinations located at $p^1 = [2, 6, 0, 0]^\top$ and $p^2 = [6, 12, 0, 0]^\top$.

The simulation result in Figure 7 shows that the executed trajectory remains similar even when the dynamic system model has been changed. That is, the system tries to enhance the backup plan safety. The path is not smooth compared to those of the double integrator model. This is because the simple car model is a nonlinear model with non-holonomic constraints that restrict the motion, e.g., it cannot make a turn when $v_t = 0$, and cannot move to the lateral direction.

C. Analysis and Discussion

This section presents simulation results regarding computation, cost, and standard deviation.
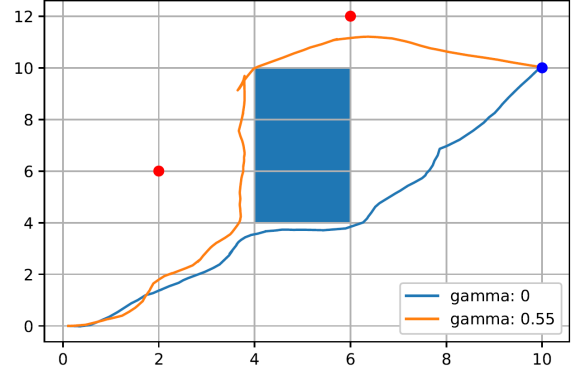

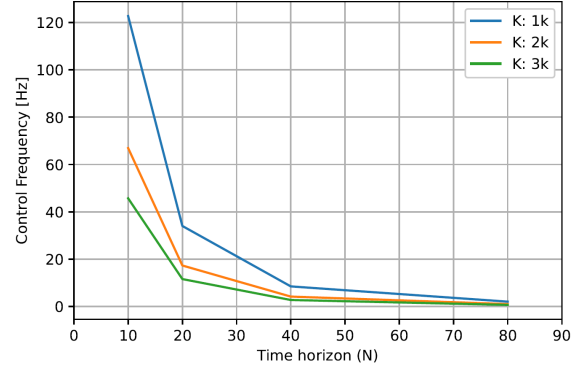Fig. 7: UGV simulation in an environment with obstacles with $N = 10$ and $K = 10000$.

Fig. 8: Control frequency with respect to the prediction horizon N.

Figure 8 shows control frequency with respect to the prediction horizon N. We have used a desktop computer with *AMD Ryzen 5 3600* CPU, 16GB RAM, and *NVIDIA GeForce RTX 2070* GPU for running the simulation with the proposed 3M algorithm. As the number of inputs increases in the order of N^2, the computational complexity for the MPPI part is expected to increase in N^2 as well. Using GPUs, it is real-time implementable up to $N = 40$.

Figure 9 shows that the average cost remains high when it is near-sighted (i.e., N is small). The cost decreases as N increases until $N = 40$. After then, the cost starts to increase because $\mathbf{J}^i$ is the sum of N horizon costs.

The standard deviation of the cost distribution decreases as K increases as shown in Figure 10. This result is consistent with the known result in statistics that the variance of importance sampling (IS)[1] estimator depends on the number of samples, i.e., $\sigma_{\text{IS est.}} = \sigma_q/K$, where σ_q is the standard deviation of the random sampling (equation (6.5) in [25]). Also, the increasing variance in N can be explained by an accumulation of variance due to the addition of random variables.

V. CONCLUSION

The motivation of this work is to enable the development of the new safety concept for autonomous systems, while

---

[1]The MPPI uses normal distribution samples to estimate the distribution of the cost of optimized control. This method of using a different distribution for estimating target distribution is called importance sampling in statistics.
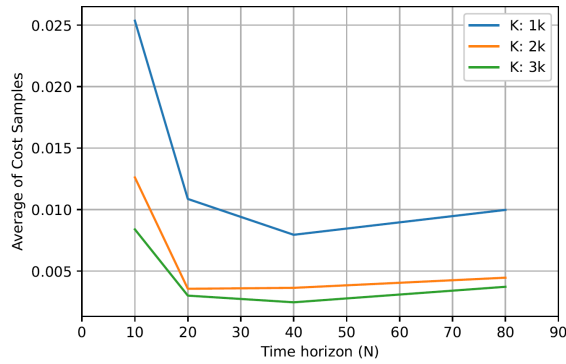
Fig. 9: Average of the cost samples $\alpha^\top \mathbf{J}$.
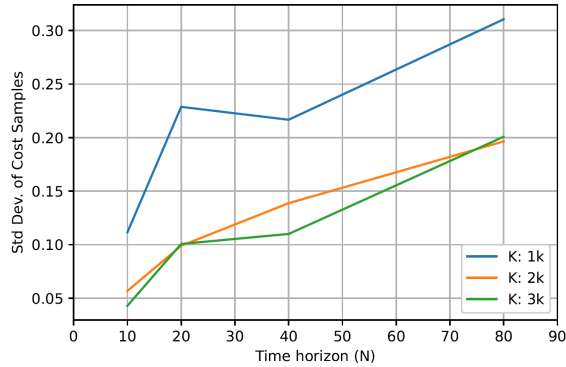

Fig. 10: Standard deviation of the cost samples $\alpha^\top \mathbf{J}$.

the current one has been limited to collision avoidance. For instance, many industries utilize the constrained motion planning on their systems and could benefit from collision-free for safety, such as indoor navigation robots, follow-filming drones, and self-driving cars. But more often than not, only considering the collision avoidance at the planning level is not enough for safety-critical systems since the primary mission may not be feasible under some unforeseen conditions. This work addresses these concerns by introducing a novel safety concept: backup plan safety that also considers the feasibility of the alternative missions. Otherwise, searching for alternative missions after finding the primary mission is not feasible could lead to dangerous consequences.

This paper studies a novel safety concept, backup plan safety. To fulfill the safety in the control problem, we formulate the control problem as a feasibility maximization problem, which is addressed by multi-horizon multi-objective model predictive path integral control, which adopts additional control horizons toward the alternative missions on top of the control horizon toward the primary mission. Simulations of aerial vehicle and ground vehicle control problems illustrate the new concept of backup plan safety and the performance of the proposed algorithms.

REFERENCES

[1] P. P. Marra, C. J. Dove, R. Dolbeer, N. F. Dahlan, M. Heacker, J. F. Whatton, N. E. Diggs, C. France, and G. A. Henkes, "Migratory canada geese cause crash of US Airways Flight 1549," *Frontiers in Ecology and the Environment*, vol. 7, no. 6, pp. 297–301, 2009.
[2] J. A. DeSantis, "Engines turn or passengers swim: A case study of how ETOPS improved safety and economics in aviation," *Journal of Air Law and Commerce*, vol. 78, p. 3, 2013.
[3] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.
[4] S. Aggarwal and N. Kumar, "Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges," *Computer Communications*, vol. 149, pp. 270–299, 2020.
[5] A. Aswani, H. Gonzalez, S. S. Sastry, and C. Tomlin, "Provably safe and robust learning-based model predictive control," *Automatica*, vol. 49, no. 5, pp. 1216–1226, 2013.
[6] L. Hewing, J. Kabzan, and M. N. Zeilinger, "Cautious model predictive control using Gaussian process regression," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2736–2743, 2019.
[7] A. Bemporad and D. M. de la Peña, "Multiobjective model predictive control," *Automatica*, vol. 45, no. 12, pp. 2823–2830, 2009.
[8] J. Hu, J. Zhu, G. Lei, G. Platt, and D. G. Dorrell, "Multi-objective model-predictive control for high-power converters," *IEEE Transactions on Energy Conversion*, vol. 28, no. 3, pp. 652–663, 2013.
[9] J. Hu, Y. Li, and J. Zhu, "Multi-objective model predictive control of doubly-fed induction generators for wind energy conversion," *IET Generation, Transmission & Distribution*, vol. 13, no. 1, pp. 21–29, 2018.
[10] F. Ascione, N. Bianco, C. De Stasio, G. M. Mauro, and G. P. Vanoli, "A new comprehensive approach for cost-optimal building design integrated with the multi-objective model predictive control of HVAC systems," *Sustainable Cities and Society*, vol. 31, pp. 136–150, 2017.
[11] R. Zhao, P. Wong, Z. Xie, and J. Zhao, "Real-time weighted multi-objective model predictive controller for adaptive cruise control systems," *International Journal of Automotive Technology*, vol. 18, no. 2, pp. 279–292, 2017.
[12] S. Li, K. Li, R. Rajamani, and J. Wang, "Model predictive multi-objective vehicular adaptive cruise control," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 3, pp. 556–566, 2010.
[13] M. Cannon, B. Kouvaritakis, S. V. Raković, and Q. Cheng, "Stochastic tubes in model predictive control with probabilistic constraints," *IEEE Transactions on Automatic Control*, vol. 56, no. 1, pp. 194–200, 2010.
[14] M. Cannon, Q. Cheng, B. Kouvaritakis, and S. V. Raković, "Stochastic tube MPC with state estimation," *Automatica*, vol. 48, no. 3, pp. 536–541, 2012.
[15] L. Blackmore, M. Ono, A. Bektassov, and B. C. Williams, "A probabilistic particle-control approximation of chance-constrained stochastic predictive control," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 502–517, 2010.
[16] A. L. Visintini, W. Glover, J. Lygeros, and J. Maciejowski, "Monte carlo optimization for conflict resolution in air traffic control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 4, pp. 470–482, 2006.
[17] N. Kantas, J. Maciejowski, and A. Lecchini-Visintini, "Sequential Monte Carlo for model predictive control," in *Nonlinear Model Predictive Control*, pp. 263–273, Springer, 2009.
[18] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1433–1440, 2016.
[19] G. Williams, A. Aldrich, and E. A. Theodorou, "Model predictive path integral control: From theory to parallel computation," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 344–357, 2017.
[20] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Information-theoretic model predictive control: Theory and applications to autonomous driving," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1603–1622, 2018.
[21] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*, vol. 26, no. 6, pp. 369–395, 2004.
[22] A. Konak, D. W. Coit, and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliability Engineering & System Safety*, vol. 91, no. 9, pp. 992–1007, 2006.
[23] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
[24] S. M. LaValle, *Planning algorithms*. Cambridge University Press, 2006.
[25] A. B. Owen, "Monte Carlo theory, methods and examples," 2013.