Sampling of Pareto-Optimal Trajectories using Progressive Objective Evaluation in Multi-Objective Motion Planning

Jeongseok Lee, Daqing Yi, and Siddhartha S. Srinivasa
Computer Science & Engineering, University of Washington
{jslee02,dqyi,siddh}@cs.washington.edu

***Abstract*— In this paper, we introduce a Markov chain Monte Carlo (MCMC) method to solve multi-objective motion-planning problems. We formulate the problem of finding Pareto-optimal trajectories as a problem of *sampling* trajectories from a Pareto-optimal set. We define an implicit uniform distribution over the Pareto-frontier using a dominance function and then sample in the space of trajectories. The nature of MCMC guarantees the convergence to the Pareto-frontier, while the uniform distribution ensures the diversity of the trajectories. We also propose progressive objective evaluation to increase efficiency in problems with expensive-to-evaluate objective functions. This enables determination of dominance relationship between trajectories before they are entirely evaluated. We finally analyze the effectiveness of the framework and its applications in robotics.**

I. Introduction

One metric is often insufficient to evaluate the *quality* of a robot's motion. For example, when a robot plans to grasp an object, it might simultaneously want to minimize path length (in meters), execution time (in seconds), path smoothness like motion jerk [1] (in meters per second^2), and distance to various obstacles (in meters again).

Such multiple objectives are ubiquitous. For example, in multi-agent planning [2], [3], robots might simultaneously want to minimize a *personal* objective like path-length (in meters) and maximize a *global* objective like search coverage (in meters^2) or team connectivity (potentially unitless if considering the eigenvalues of the connectivity Graph Laplacian).

Such multi-objective planning problems are often converted into single-objective problems by selecting a utility function that trades off each objective. However, tuning these functions is challenging as they are often scaling different units (like above), and the relative importance of different objectives might be quite context-dependent.

Interestingly, some solutions are categorically worse than others, no matter what utility function is chosen to compare them. For example, in Fig. 2, the gray path is worse than the green path in *both* objectives f_1 and f_2. We call the grey path *dominated*. Note, however, that neither the green nor the orange path is dominated. The set of all non-dominated paths forms the *Pareto-frontier* [4], [5], and each path in this set is *Pareto-optimal*. This leads to our central tenet:

> Providing the user samples from the Pareto-frontier gives them a choice to *select* a solution for the specific problem instead of having to *specify* a utility function apriori.

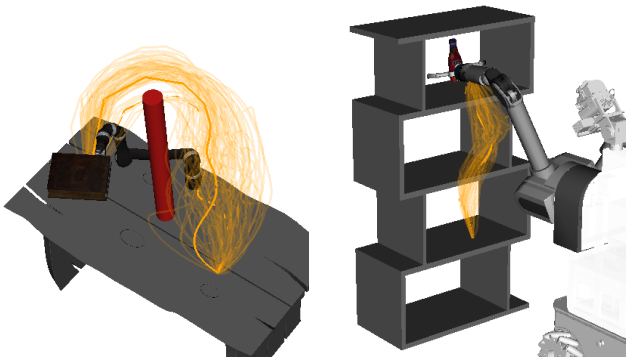
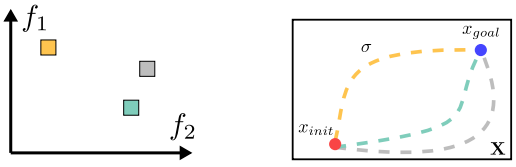


Fig. 1: The proposed framework of MOMCMC is demonstrated in finding collision-free shortest paths on table-top manipulation problems (left) and in cluttered environments like a bookshelf (right).

We now face two key challenges: (1) sampling provably *uniformly* from the Pareto-frontier, and (2) producing samples *efficiently*. Our key insight is that this can be solved by defining a uniform distribution of trajectories on the Pareto-frontier and introducing Markov chain Monte Carlo (MCMC) to efficiently generate trajectory samples. In this paper, we proposed a framework of Multi-Objective MCMC (MOMCMC) in uniformly and efficiently sampling Pareto-optimal trajectories (Sec. IV). The framework of MOMCMC includes (1) the definition of a target distribution and (2) Markov chain random walks that generate new samples. The target distribution is defined by a dominance function (Sec. V) that measures how a trajectory is dominated by others. We also included simulated anneal in tuning the temperature parameter in the target distribution that gradually transforms the target distribution toward a uniform distribution on the Pareto-frontier. We propose two types of Markov chain random walks, differential evolution and fast non-dominated sorting (Sec. V). Because evaluating the objective of a trajectory is a progressive accumulation of the cost along the trajectory, the expense of evaluating objective can become arbitrarily high and a bottleneck in performance. We introduce progressive objective evaluation (in Sec. VI) to reduce unnecessary evaluation in determining the dominance of trajectories.

In our experiments in Sec. VII, we compared a few random walk methods in the MOMCMC framework and a classic



(a) Objective space. (b) Configuration space.

Fig. 2: Motion planning with multiple objectives. Each point in the objective space (a) maps to a trajectory in the configuration space (b).

multi-objective optimization solver MOEA/D [6]. We used the motion-planning of a drone to demonstrate that the methods in the MOMCMC framework generate Pareto-optimal trajectories with good diversity and are also scale-invariant in problems with two objectives to seven objectives. We evaluated the performance of progressive objective evaluation in different problem settings. We also demonstrated how the MOMCMC framework can be robustly and efficiently applied to re-planning and bi-arm manipulation problems.

Our major contribution is the proposal of the framework of Multi-Objective Markov Chain Monte Carlo in efficiently and uniformly sampling Pareto-optimal trajectories and two methods of random walks. We demonstrated the performance of the framework in generating trajectories in different planning problems, planar-robot navigation and bi-arm manipulation.

II. Related Work

Our work supports multi-objective optimization [7] in the motion-planning field, and our algorithms operate in continuous trajectory parametric space. Previous algorithms in planning usually run on an existing discrete graph structure, so Multiobjective A* [8] applies to the optimal search for motion planning. If paths on a discrete structure could be encoded into a vector representation, e.g., a sequence of directions in a grid world [4] and B-spline with a fixed number of control points [9], then many evolutionary multi-objective solvers could be introduced to find a trajectory. An alternative approach is sophistically building a discrete structure of sampled states in a workspace by multi-objective heuristics to facilitate the search. MORRF* [5] creates multiple rapidly-exploring random trees in parallel to find a set of Pareto-optimal paths.

Our work is also related to optimization in trajectory space in motion-planning problems. CHOMP [10] introduces trajectory costs that are invariant to time parameterization of trajectories to solve the motion-planning problem in a parametric trajectory space. It inspires STOMP [11] with stochastic optimization and ITOMP [12] in re-planning. TrajOpt [13] solved the corresponding motion-planning problem using sequential convex optimization so that the returned solution will converge to a local optimal. CHOMP [10] introduces the idea of sampling trajectories from a distribution defined by trajectory cost following Hamiltonian Monte Carlo so that a global optimal can be found. Cross-entropy motion planning extends this idea into cross-entropy sampling [14], and iteratively updating the distribution so that sampled trajectories converge to the optimal. In this paper, we extend sampling trajectories into a Pareto optimal set so that all sampled trajectories are Pareto-optimal.

Evolutionary computation is dominating in solving multi-objective optimization problems because it well applies to discontinuous, non-differential, multi-modal and not well-defined problems [7]. Different algorithms have been proposed to evolve a population of solutions toward a Pareto-frontier by metaheuristic. NSGA-II [15] sorts solutions by non-dominance evaluation; MOEA/D [6] decomposes a multi-objective optimization problem into a set of sub-problems with different sampling weights. Considering the diversity of solutions, sampling methods in Monte Carlo simulation are introduced to generate solutions from a distribution defined for a Pareto-frontier. MOMCMC [16] defines a cost function that measures non-dominances, which implies a target distribution over Pareto-frontier. MCMC is applied to generate samples that uniformly distributed on the Pareto-frontier. Also in a sampling approach, cross-entropy method [17], [18] is introduced to generate samples from a distribution over the Pareto-frontier.

Extending multiple metrics and constraints into multiple objectives helps efficiency in the expensive evaluation. POMP [19] extends collision checking into a new objective that iteratively updates a belief model of collision while searching on a Pareto-frontier. It is proved that the number of collision checking is efficiently reduced.

III. Problem Definition

We define a trajectory, $\sigma : [0,1] \; ! \; C \quad \mathbb{R}^D$ is a continuous function that maps time $t \; 2 \; [0,1]$ into a configuration $q \; 2 \; C$ (Fig. 2b). An objective is a functional that returns a real value for a trajectory σ. When there are multiple objectives, we use a vector of objective functionals $f(\sigma) = [f_1(\sigma), \quad , f_K(\sigma)]^T$. Each trajectory σ can be represented as a point in an objective space where each dimension corresponds to a particular objective $f_i(\sigma)$ (Fig. 2a).

Between any two trajectories, there are three types of relations by trajectory dominance in an objective space.

dominate $\sigma_1 \quad \sigma_2$: σ_1 dominates σ_2 when $8i, f_i(\sigma_1) < f_i(\sigma_2)$.

dominated $\sigma_1 \quad \sigma_2$: σ_1 is dominated by σ_2 when $8i, f_i(\sigma_1) > f_i(\sigma_2)$. This is equivalent to $\sigma_2 \quad \sigma_1$.

parallel $\sigma_1 \; k \; \sigma_2$: σ_1 is parallel to σ_2 when neither of them is smaller than the other in all the objectives.

For example, in Fig. 2b, the green trajectory *dominates* the gray trajectory, and the yellow trajectory is *parallel* to both the gray trajectory and the green trajectory. The "quality" of a trajectory can be measured by its relations with other trajectories in an objective space.

Our goal is to find a set of Pareto-optimal trajectories, which is also known as "non-dominated trajectories" [15] in an objective space. A non-dominated trajectory implies that no other trajectory "dominates" it. It means any two non-dominated trajectories are "parallel" to each other.

IV. Uniform Sampling of Pareto-optimal Trajectories

We solve a multi-objective motion-planning problem by defining a uniform distribution of trajectories on an implicit Pareto-frontier. This method belongs to a family of trajectory sampling algorithms [14], [10]. We use a sampling approach to produce trajectories from the distribution, which is inspired from Multi-Objective Markov Chain Monte Carlo (MOMCMC) [16]. The distribution is constrained to a Pareto-frontier so the sampled trajectories are Pareto-optimal. The uniformity of the distribution assures the diversity of sampled trajectories.

We first define a Gibbs distribution

$$P(x) \;/\; \exp\left(\; f_{dom}(x)/T\right)$$

where $f_{dom}()$ is a dominance function (non-negative "energy function") and T denotes a "temperature" parameter. A dominance function $f_{dom}()$ measures how much a trajectory is dominated by other trajectories. We want to design a dominance function so that the corresponding distribution is an uniform distribution over the Pareto-frontier. Such a dominance function should satisfy following properties as follows:

$f_{\mathrm{dom}}(\sigma) = 0 \;\; \forall \sigma \in$;
$f_{\mathrm{dom}}(\sigma) > 0 \;\; \forall \sigma \notin$; and
$\sigma_1 \in \;\; \sigma_2 \notin \;, \; f_{\mathrm{dom}}(\sigma_1) < f_{\mathrm{dom}}(\sigma_2) \;\; \forall \sigma_1, \sigma_2 \in$.

where denotes a Pareto-optimal set in a trajectory space. All the non-dominated trajectories have the same value of $f_{\mathrm{dom}}(\sigma)$ thus the same probabilities, which implies uniformity as Property 1.

Property 1: $P(\sigma \mid f_{\mathrm{dom}}(\sigma) = 0)$ is a uniform distribution, which is $\forall \; _1, \; _2 \in \;, P(\; _1 \mid f_{\mathrm{dom}}(\; _1) = 0) = P(\; _2 \mid f_{\mathrm{dom}}(\; _2) = 0)$.

Tuning the parameter T also reshapes the distribution. As T approaches to zero, the probability of any dominated trajectory converges to zero. At the same time, the probability of any non-dominated trajectory is increased. We have that $P(\sigma)$ converges to a uniform distribution on the Pareto-optimal set $P(\sigma \mid f_{\mathrm{dom}}(\sigma) = 0)$ as $T \to 0$ in Property 2.

Property 2: $\lim_{T \to 0} P(\sigma) = P(\sigma \mid f_{\mathrm{dom}}(\sigma) = 0)$.

We describe the framework of sampling trajectories in Algorithm 1 which uses Markov Chain Monte Carlo. We first initialize a population of trajectories from an arbitrary known (e.g., uniform) distribution in a trajectory space . Driven by Markov chain random walks, all the samples gradually converge to a target distribution. Metropolis correction (line 8 13 in Algorithm 1) plays the pivotal role in assuring the detailed balance of MCMC [20]. One of the virtues of the Metropolis correction is that it allows selecting any random walk that possibly speeds up the convergence to the target distribution. Denote i-th trajectory in the current population as $[i]$. Reaching Line 11 means that the new i-th trajectory σ_i' is accepted thus increasing the acceptance count N_{acc}. Reaching Line 13 means that the new i-th trajectory σ_i' is rejected thus is reverted to σ_i in a previous population. As the acceptance rate increases, temperature T is decreased that gradually moves the distribution toward $P(\sigma \mid f_{\mathrm{dom}}(\sigma) = 0)$, which is a uniform distribution over Pareto-frontier.

Algorithm 1 MOMCMC $(P(\sigma \mid f_{dom}), N, \mathbf{f}, \gamma_{acc})$

```
1:      UNIFORM( , N)
2:  while NOTCONVERGENCED( ) do
3:      f_Dom   FITNESSEVALUATION( , f)
4:      N_acc   0
5:       '   MARKOVCHAINRANDOMWALK( )
6:      f'_Dom   FITNESSEVALUATION( _i, f)
7:      for i = 1 to N do
8:          u   UNIFORM([0, 1), 1), σ_i   [i], σ'_i   '[i]
9:          f_dom(σ_i)   f_Dom[i], f_dom(σ'_i)   f'_Dom[i]
10:         if u < exp( (1/T)(f_dom(σ'_i)   f_dom(σ_i))) then
11:             N_acc   N_acc + 1
12:         else
13:             σ'_i   σ_i
14:             '
15:     if N_acc/N < γ_acc then
16:         INCREASE(T)
17:     else
18:         DECREASE(T)
    return
```

The fitness of a trajectory is evaluated by f_{dom}. We thus can evaluate the fitness of a population of trajectories in FITNESSSEVALUATION(). The performance of Algorithm 1 is determined by (i) how the dominance is evaluated FITNESSSEVALUATION(); and (ii) how the Markov chain random walk is designed MARKOVCHAINRANDOMWALK(). We will provide one form of FITNESSSEVALUATION() in Section V. We will discuss the design of MARKOVCHAINRANDOMWALK() that could converge faster to a target distribution.

V. Dominance Function and Markov Chain

We begin with decomposing an objective space for a given trajectory σ (Fig. 3). Given an arbitrary trajectory σ, we can decompose a trajectory space into three disjoint subsets by the three types of relations as in Fig. 3:

$= \{\sigma' \in \;\; \sigma' \prec \sigma\}$: The subset of all trajectories dominate σ

$= \{\sigma' \in \;\; \sigma \prec \sigma'\}$: The subset of all trajectories dominated by σ

$_k = \{\sigma' \in \;\; \sigma' \parallel \sigma\}$: The subset of all trajectories that are parallel to σ

We find that the volume of , written as $|\quad|$, satisfies the three properties of $f_{\mathrm{dom}}()$. $|\quad| = 0$ if and only if a trajectory σ is non-dominated. If σ_1 dominates σ_2, then $|\quad _1| < |\quad _2|$. Thus we design a dominance function $f_{\mathrm{dom}}(\sigma)$ for trajectory σ by "estimating" the volume of .

In practice, it is impossible to analytically calculate $|\quad|$ for any given σ. We propose $\hat{f}_{dom}(\sigma \mid \;\;)$ as an estimation of $f_{\mathrm{dom}}(\sigma)$ for trajectory σ based upon a population in (1), which is a variation of the fitness function introduced in

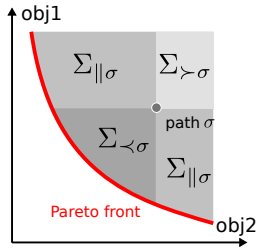


Fig. 3: A trajectory space is decomposed by dominance relationships given a trajectory σ.

[16], [17]:

$$\hat{f}_{\mathrm{dom}}(\sigma j \quad) = \mathsf{P} \quad \begin{matrix} s(\sigma) \quad 1 & \sigma\ 2 \\ , \quad \mathrm{HYP}(\sigma, \sigma^0) & \sigma\ \not\in \end{matrix} \quad . \tag{1}$$

$s(\sigma)$ indicates the proportion of trajectories in the population that are dominated by σ. $\mathrm{HYP}(\sigma, \sigma^0)$ is defined by the hypervolume between σ and σ^0 using Lebesgue measure, which is associated with the significance of dominance. Note that $\hat{f}_{dom}$ satisfies the properties of f_{dom}, except non-zero fitness is also used for non-dominated trajectory [16], [17] to facilitate Markov chain random walk on a Pareto-frontier. Notice that (1) returns non-positive value when a trajectory is non-dominated in the population, instead of zero. In practice, it will better support random walk on a Pareto-frontier for better diversity in MCMC.

Following the definition of $\hat{f}_{dom}$, we use differential evolution [21], [16] to generate new samples in MARKOVCHAINRANDOMWALK. We also propose another type of MARKOVCHAINRANDOMWALK, which uses fast non-dominance sorting [15]. In this approach, we use tournament selection that selects from current population to produce new candidates. The new candidates will be merged into current population. Fast non-dominance sorting is then called to sort the population. The top samples in the population became the new population, which is equivalent to new samples produced in Markov chain random walk.

VI. PROGRESSIVE OBJECTIVE EVALUATION

The computational cost of an objective increases proportionally to the cost of the integrand. This can be critical when we have an expensive integrand function such as collision penalty [22], which is common and critic in motion planning. One beneficial property of this kind of objectives is that it is a monotonically increasing functional if the cost function for all the configuration is non-negative. This property motivates us to propose a progressive objective evaluation to reduce unnecessary evaluation in determining dominance relations.

Defining $\sigma(t)$ as the segment of a trajectory σ in $[0, t]$, we can have $f_k(\sigma(t)) \quad f_k(\sigma), \forall k$. By transitivity, we have if $f_k(\sigma_1) \quad f_k(\sigma_2(t))$ then $f_k(\sigma_1) \quad f_k(\sigma_2)$. Extending it to determine dominance, we have $t\ 2\ [0,1] \wedge \sigma_1 \quad \sigma_2(t) \ ) \ \sigma_1 \quad \sigma_2$. It tells that we can know whether a trajectory σ_1 dominates σ_2 before fully evaluating the whole σ_2. We propose a *progressive objective evaluation* in Algorithm 2, which evaluates multiple objectives progressively. Evaluated objectives are then used to calculate the fitness in Eq. (1).

Algorithm 2 defines a process of progressively evaluating the objectives of trajectories in a population. The progress of evaluating a trajectory depends on the dominance relations with other trajectories in the population and their current evaluations. It prevents unnecessary time cost in evaluating objectives once dominant relations (i.e., *dominating* or *dominated*) can be determined. Each trajectory has K objectives to evaluate separately and sequentially.

We introduce a few matrices to track the progress of evaluations. We first define a matrix T to store the progresses of evaluations and a matrix O to store the current evaluated objective values. $\mathsf{T}(i,k)$ records the progress of evaluating k-th objective of i-th trajectory. $\mathsf{O}(i,k)$ records the evaluated value in k-th objective of i-th trajectory. By tracking the comparison between evaluated trajectories, we can determine whether an early termination is allowed. We use a matrix S to store the status of evaluation up k-th objective evaluation and a matrix E to store whether evaluation shall continue. $\mathsf{S}(i,j)$ indicates the relation of i-th trajectory to j-th trajectory. There are four evaluation statuses, D, d, p and u. d, D, and p corresponds to three types of relations defined in Section V, which are *dominate*, *dominated*, and *parallel*, respectively. u is newly added here to represent an "undecided" relation. The value in $\mathsf{S}(i,j)$ determines the value in $\mathsf{E}(i,j)$, which is a boolean value indicating whether evaluating $\mathsf{S}(i,j)$ shall be continued. There is also an ordered array I that stores the indices of trajectories to be evaluated that is sorted in the descending order of the current objective values, and an ordered array C that stores trajectories whose objective evaluation is entirely evaluated.

Algorithm 2 calls PROGRESSEVAL(σ_i, k) in evaluating k-th objective of i-th trajectory σ_i progressively, which returns the portion that has been evaluated $\mathsf{T}(i,k)$ and the corresponding objective $\mathsf{O}(i,k)$. Because objectives are evaluated by sequence, we introduce $\quad_k$ to denote that a path dominates the other in first k-th objectives. The high-level idea behind Algorithm 2 can be summarized as three cases as

1) $\mathsf{O}(i,k) < \mathsf{O}(j,k) \wedge \mathsf{T}(i,k) = 1 \wedge \mathsf{T}(j,k) < 1$ $) \ f_k(\sigma_i) < f_k(\sigma_j)$;
2) $f_k(\sigma_i) < f_k(\sigma_j) \wedge \sigma_i \quad_k \ _1 \ \sigma_j \quad) \quad \sigma_i \quad_k \sigma_j$;
3) $f_k(\sigma_i) < f_k(\sigma_j) \wedge \sigma_j \quad_k \ _1 \ \sigma_i \quad) \quad \sigma_i \ k \ \sigma_j$.

DOMUPDATE() is called when we find that one trajectory is less than another in one objective. At the end, it calls FITNESSEVAL() that takes in evaluated objectives O to generate the fitnesses of trajectories.

VII. EXPERIMENTS

In our experiments, we sample in a trajectory parametric space S [10], instead of directly sample in a trajectory space . A trajectory parametrization defines a map between a trajectory σ in an infinite trajectory space and a parameter s in a finite trajectory parametric space S. The parametric-invariance of the objective functionals enables that we obtain consistent trajectories independent on parameterization methods [23]. The objective is evaluated by an integral that accumulates the non-negative cost along the trajectory, which

Algorithm 2 PROGRESSIVEFITNESSEVALUATION (; f)

```
O( ; )   0, T( ; )   0, I   , C   ;
S( ; )   u, E( ; )   true
for k   1 to jf j do
    8i; E(i; i )   false
    for  i ;  j 2   do
        if S(i; j ) = p then
            E(i; j )   false
    while I 6= ; do
         i   FRONT(I )
        O(i; k ); T(i; k )   PROGRESSEVAL( i ; k)
        for  j 2 C do
            if  E(i; j ) = false ^ S(i; j ) = p  then
                continue
            if  O(i; k ) > O(j; k )  then
                S; E   DOMUPDATE(i; j; S; E)
            else if O(i; k ) < O(j; k ) ^ T(i; k ) = 1  then
                S; E   DOMUPDATE(j; i; S; E)
            if E( ; j ) = false then
                C   C n f j g
        if T(i; k )   1 then SORT(I ; k)
        else
            I   I n f  ig, C   C [f  ig
            for  j 2 I  do
                if O(i; k ) < O(j; k ) ^ T  then
                    S; E   DOMUPDATE(j; i; S; E)
    I   , C   ;
    for  i 2 I  do
        if S( ; i) = p then
            I   I n f  i g
return FITNESSEVAL(O)
```

Algorithm 3 DOMUPDATE (i; j; S; E)

```
if S(i; j ) = D  then
    S(i; j )   S(j; i )   p, E(i; j )   E(j; i )   false
else if S(i; j ) = u then
    S(i; j )   d, S(j; i )   D , E(i; j )   E(j; i )
false
```

is $f(\ ) = \int_R c(\ (t))\|\frac{d}{dt}x(\ (t))\|dt$. The definition of objective functional guarantees that the objective is parametric-invariant [10], which means that the results returned from a solver are independent of the type of parametrization method.

We choose cubic B-splines in trajectory parametrization, $q(t; x) = \sum_{i=0}^{3} B_i(t)x_i$, where $B_i(t)$ are the spline basis functions and x_i are the control points. We use four control points for each trajectory where the first and end points are fixed, and the two middle points are free. This is to make the start and goal configurations fixed. B-spline has properties that (1) joint position limits are straightforwardly enforced using the convex hull property [24]; and (2) joint velocity limits can be formulated as linear constraints [24]. A sample s is obtained in a trajectory parametric space S. The sample is mapped into a trajectory in evaluating the objective.

(a) Trajectories in objective space.

(b) Percent of non-dominated trajectories.

(c) Diversity in objective space.

Fig. 4: planning for a drone with a path-length objective and an informative objective using different approaches.

We test the performance of the algorithms in two problems, (1) a drone in a search task; and (2) a robot HERB [25] working on a table using both arms. The performance of the algorithm is evaluated by Pareto-optimality and diversity of samples The Pareto-optimality is measured by non-dominance to an elite set (i.e., the non-dominated samples in a population) because the true Pareto-optimal set is impossible to evaluate. The diversity of samples is measured by spread[15],

$$= \frac{\sum_{k=1}^{K} d_k^e + \sum_{i=1}^{N} |d_i - d|}{\sum_{k=1}^{K} d_k^e + N d} \qquad (2)$$

given a population of N samples, d_i is the distance between i-th sample to its nearest neighbor, d is the mean of all the d_i, and d_k^e is the distance between the extreme solutions of the true Pareto-frontier and the population on k-th objective. The larger the spread is, the better diversity of the samples is. We test three different methods of random walks under the framework of MOMCMC, which are MOMCMC using Differential Evolution (DE), MOMCMC using progressive Differential Evolution (PROG-DE) and MOMCMC using Fast Non-dominance Sorting (FNS). We also compare them with MOEA/D solver [6]. We implement our framework and experiments on top of AIKIDO [26] that uses DART [27] for the underlying rigid body simulation and visualization.

A. Performance

Our first experiment uses a single drone with two objectives, which are minimizing path length and maximizing information gain. The information gain is measured by the accumulated information along with a trajectory, in which the information in each configuration is defined by a distribution of information.

Fig. 4 shows the results of different approaches with the same number of iterations, 200. It is evident that in the objective space trajectories by MOEA/D shows the lower

diversity, as in Fig. 4a. The reason is that MOEA/D uses a set of random weights to generate subproblems to solve [6], the scale difference distorts uniformly sampled weights mapping to clustered solutions. Fig. 4c indicates three approaches of MOMCMC provide close performance in diversity, and FNS outperforms the others. Fig. 4b shows that FNS and MOEA/D return more non-dominated trajectories, which indirectly reflects the convergence rate.

We then test the scale-invariance of the approaches. Scale-invariance measures whether algorithms still generate consistent results if some of the objectives are scaled. Fig. 5 shows the results in a two-objective problem. We manually scale the second objective by multiplying different scaling factors w (0.01 and 100). We can tell that all three approaches in MOMCMC (DE, PROG-DE, and FNS) provides similar trajectories as in Fig. 5, while the trajectories by MOEA/D are dramatically changed. It indicates that the approaches in MOMCMC are robust to how units are selected for objectives and can provide consistent results subject to the scaling difference. The performance of MOEA/D is not ideal in this example, due to its weight-based nature discussed above [6].

We test three approaches of MOMCMC with a different number of objectives. We want to verify the performance of the approaches adapt to the increasing number of objectives. We take ten runs on each problem in collecting data to verify the performance robustness. Each run is with 1,000 iterations. Fig. 6a shows all three approaches have consistent performance on diversity when more objectives are introduced. FNS is consistently better in diversity than others in Fig. 6a. All three approaches show close performance in generating non-dominated trajectories in Fig. 6b.

Fig. 7 shows the performance of the progressive objective evaluation with different problem size and evaluation step sizes. RES80 indicates less number of steps to evaluate than RES100, which means a larger step size. Fig. 7a and 7b show that the time saved by PROG-DE is proportional to the portion of path evaluation. Fig. 7a shows that reducing the evaluation step size does not impact the performance. Increasing the problem size will diminish the time saved from progressive objective evaluation because it becomes harder to significantly dominate since there are more other samples in competing dominance.

B. Re-planning

The chain nature of the MOMCMC framework makes it a good fit for re-planning under environment changes. Assume that the new target distribution on a new Pareto-frontier is caused by environmental change is not too different from the previous one used for sampling. The solutions from the initial planning could be used as the initial samples used in MCMC in the re-planning, we call this a *warm re-planning*. Instead, we call it a *cold re-planning* if it is an MCMC from scratch. From statistics, we know that it takes much less burn-in iterations to converge to the new target distributions using a warm re-planning than using a cold re-planning. We run both types of re-planning with 10 iterations only. Fig. 8a shows an initial set of Pareto-optimal trajectories from previous planning. The information source is moved from $(5.0, 0.0)$ to $(7.5, 0.0)$ both in Fig. 8b and 8c, which implies the same new Pareto-frontier. We can see that in Fig. 8c a warm re-planning almost converges trajectories to a new Pareto-frontier in 10 iterations, while a cold re-planning is far away from convergence in Fig. 8b. The difference can also be told in an objective space in Fig. 8d.

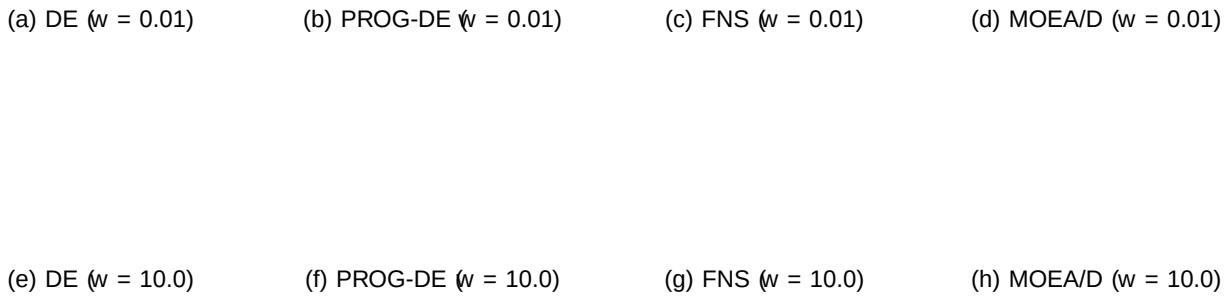
C. Real-world Problem

We also test our framework in a bi-arm manipulation planning problem with HERB [25]. The task is swapping two bottles on a table using two-arm manipulation. The start pose and goal pose are shown in Fig. 9a and 9b. We consider two arms as two individual agents, so that have their objectives while they coordinate to maximize team objectives. We choose five objectives in this experiment, which are configuration space path lengths for the arms (two objectives), collision cost between two arms, and collision cost between the table and each arm (two objectives). The path length is measured by integrating the trajectory as a consecutive 50 line segments. We consider hard constraint, such as collision, as one objective by using a penalty functional. We accumulate penalty of 1.0 whenever the discretized trajectory is in a collision so that zero value means collision-free. For each trajectory, the dimension of parametric trajectory space is 28 (i.e., two 7-DOF arms $\times$ two free control points). In Fig. 9 we only show the results to compare DE and FNS in 500 iterations. FNS shows faster convergence so that there are more non-dominated trajectories returned from FNS, as shown in Fig. 9c. Fig. 9d and 9e visualize the sampled trajectories by DE and FNS. Since the trajectories sampled by DE are not yet converged to the Pareto-frontier, it shows better diversity in Fig. 9c.

VIII. Conclusion

We propose a framework of Multi-Objective Markov Chain Monte Carlo (MOMCMC) that uniformly samples trajectories on a Pareto-frontier in solving multi-objective path-planning problems. We define a type of target distribution for MCMC to converge from arbitrary initial trajectories toward the Pareto-optimal set. We also propose two approaches of random walks for the framework to enhance the convergence rate. We conduct experiments to verify the performance of the algorithms, and how it applies to re-planning and real-world problems.

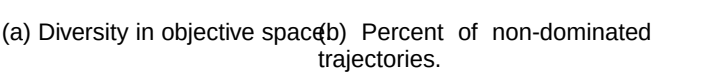
Finding a set of Pareto-optimal trajectories for multiple objectives is a more expensive approach than finding only a single trajectory. However, having a pool of candidate solutions to select make this approach robust to changes and uncertainties. A set of Pareto-optimal trajectories also recovers more information in both trajectory space and objective space, which support diversity in programming trajectory selection specifications [28], [29]. In this paper, constraints are modeled as objectives, which could be extended to solve "minimum constraint removal" in finding a feasible solution [30] by searching a feasible one in a set.



(a) DE (w = 0.01) (b) PROG-DE (w = 0.01) (c) FNS (w = 0.01) (d) MOEA/D (w = 0.01)

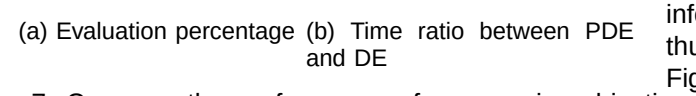
(e) DE (w = 10.0) (f) PROG-DE (w = 10.0) (g) FNS (w = 10.0) (h) MOEA/D (w = 10.0)

Fig. 5: Compare scale invariance of the approaches in a two objective problem. w indicates a scaling factor applied to 2nd objective.



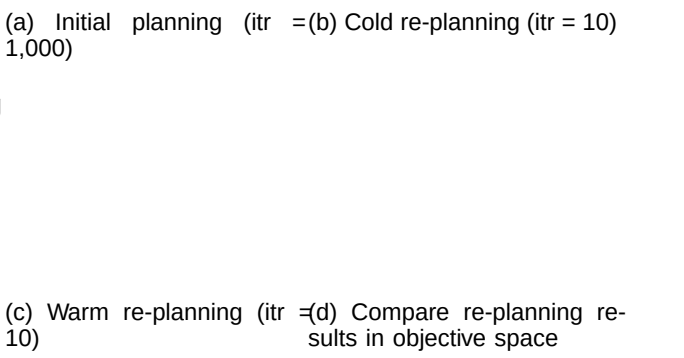
(a) Diversity in objective space (b) Percent of non-dominated trajectories.

Fig. 6: Compare different approaches and methods using different number of objectives.



(a) Evaluation percentage (b) Time ratio between PDE and DE

Fig. 7: Compare the performance of progressive objective evaluation with different problem size and different evaluation step size.



(a) Initial planning (itr = 1,000) (b) Cold re-planning (itr = 10)

(c) Warm re-planning (itr = 10) (d) Compare re-planning results in objective space

Fig. 8: Planning trajectories from the red point to the blue point. The green point indicates the location of the information sources. It is changed after planning in Fig. 8a thus two different re-planning are triggered in Fig. 8b and Fig. 8c with 10 iterations.

ACKNOWLEDGMENTS

This work was (partially) funded by the National Institute of Health R01 (#R01EB019335), National Science Foundation CPS (#1544797), National Science Foundation NRI (#1637748), the Office of Naval Research, the RCTA, Amazon, and Honda.

REFERENCES

[1] A. Piazzi and A. Visioli, "Global minimum-jerk trajectory planning of robot manipulators," IEEE transactions on industrial electronics, vol. 47, no. 1, pp. 140–149, 2000.

[2] M. Bennewitz, W. Burgard, and S. Thrun, "Optimizing schedules for prioritized path planning of multi-robot systems," in Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, vol. 1. IEEE, 2001, pp. 271–276.

[3] D. Yi, M. A. Goodrich, and K. D. Seppi, "Informative path planning with a human path constraint," in Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on. IEEE, 2014, pp. 1752–1758.

[4] F. Ahmed and K. Deb, "Multi-objective optimal path planning using