

# Predictive Optimal Control with Data-Based Disturbance Scenario Tree Approximation

Ran Jing and Xiangrui Zeng

**Abstract**—Efficiently computing the optimal control policy concerning a complicated future with stochastic disturbance has always been a challenge. The predicted stochastic future disturbance can be represented by a scenario tree, but solving the optimal control problem with a scenario tree is usually computationally demanding. In this paper, we propose a data-based clustering approximation method for the scenario tree representation. Differently from the popular Markov chain approximation, the proposed method can retain information from previous steps while keeping the state space size small. Then the predictive optimal control problem can be approximately solved with reduced computational load using dynamic programming. The proposed method is evaluated in numerical examples and compared with the method which considers the disturbance as a non-stationary Markov chain. The results show that the proposed method can achieve better control performance than the Markov chain method.

## I. INTRODUCTION

Optimal control has been applied in many control applications for systems with uncertain disturbance. The disturbance considered in this paper can be generalized to any variables in the system dynamics equations which appear in the form of disturbance. The actual physical meanings of these variables may be noise, external input, or environmental signals. For example, the driver's power demand may be considered as generalized disturbance to the vehicle, and the ambient temperature may also be considered as generalized disturbance to the building heating, ventilation, and air conditioning systems. Some disturbance may have high impacts on the optimal control cost, and the optimal control policies may depend on the characteristics of the disturbance [1].

In real-time optimal control problems, future disturbance information may be critical to the control policy. Therefore in many applications, the disturbance is predicted and utilized to generate the control inputs. Usually the disturbance is considered as a random variable, and the future disturbance is considered as a stochastic process. There are a few ways to integrate the predicted

stochastic disturbance information with the optimal control. A common practice is to consider the disturbance as a Markov chain [2] [3] [4] [5]. This representation can capture the first-order characteristics of the disturbance, but it is not capable of extracting long-term information on a larger time scale. Another way is to build a dynamic model for the stochastic system whose output is the disturbance. Then additional states describing the dynamics for disturbance are added to the existing model of the system for optimal control calculation. A disadvantage of this state augmentation method is the high computation if the disturbance is generated by a high-dimensional complex system. Scenario trees can also be used to describe the predicted disturbance information [6] [7] [8]. A tree structure is used to represent all the possible disturbance sequences in the future. The optimal control with a disturbance scenario tree can be computed using dynamic programming, which will be shown in this paper.

In general, to solve optimal control problems, methods like nonlinear programming [9], quadratic programming [10], and dynamic programming [11] [12] can be used. When the future disturbance is described as a scenario tree, dynamic programming is usually the only practical method to compute the global optimal policy. However, dynamic programming suffers from the curse of dimensionality, which means it can only be used to solve problems with a relatively small state space size. For the optimal control problem with a disturbance scenario tree, the state space size will grow exponential as the size of the tree grows. Therefore computing the predictive optimal control policy for a complicated scenario tree is very difficult. Approximate dynamic programming can be applied to find a sub-optimal solution in many optimal control problems [13]. However a good approximated method for dynamic programming with scenario trees is not yet available.

In this work, we aim at finding the approximate global optimal control policy given a scenario tree of predicted future disturbance. We propose a novel clustering-based approximation of the scenario tree and design optimal predictive policies based on this approximation. Differ-

The authors are with the Department of Robotics Engineering, Worcester Polytechnic Institute, Worcester, MA 01609 USA {rjing, xzeng2}@wpi.edu. Corresponding author: Xiangrui Zeng.

ently from the first-order Markov chain approximation, the scenario tree approximation can use information from the starting step to the current step. With the tree approximation, we can find the global optimal policy via dynamic programming with reduced computational load. Our work can also be embedded in the receding horizon framework to solve the optimal control problem in one step of model predictive control.

The rest of this paper is organized as follows. The problem formulation is given in Section II. The exact scenario tree solution and a clustering-based approximation solution are proposed in Section III and Section IV, respectively. Section IV provides numerical examples. We conclude the paper in Section VI.

## II. PROBLEM FORMULATION

### A. Predictive Optimal Control with Disturbance

Consider the discrete-time dynamic system

$$x_{k+1} = f_k(x_k, u_k, w_k), \quad (1)$$

where the system state  $x_k \in X \subseteq \mathbb{R}^n$ , the control input  $u_k \in U(x_k) \subseteq \mathbb{R}^m$ , the generalized disturbance  $w_k \in \mathbb{R}^q$ , and finite discrete time step  $k \in K = \{0, 1, \dots, N-1\}$ . The initial condition  $x_0$  is known.

The system has a running cost function

$$l_k = g_k(x_k, u_k, w_k) \quad (2)$$

and a terminal cost function

$$l_N = g_N(x_N), \quad (3)$$

where  $l_k \in \mathbb{R}$  and  $l_N \in \mathbb{R}$ . The total cost is the sum of the running cost at each step and the terminal cost,

$$l = \sum_{k=0}^{N-1} l_k + l_N. \quad (4)$$

The goal is to find control policies  $\pi_k$  which minimize the total cost  $l$ . The control policies need to be causal, which means that they can only use current and past information. It is assumed that the values of  $x_k$  and  $w_k$  are known at step  $k$ , then

$$u_k = \pi_k(x_0, \dots, x_k, w_0, \dots, w_k). \quad (5)$$

If  $w_0, \dots, w_{N-1}$  are known at step 0 (future values of the disturbance are known), this optimal control problem can be easily solved via dynamic programming. In most applications, knowing the future values of the disturbance is not possible. Many predictive control applications use predicted nominal values of the future disturbance  $\hat{w}_0, \dots, \hat{w}_{N-1}$  to solve the optimal control problem. With the predicted nominal disturbance, the

nominal optimal policy can be obtained using dynamic programming backward induction,

$$J_k(x) = \min_{u \in U(x)} \{g_k(x, u, \hat{w}_k) + J_{k+1}[f_k(x, u, \hat{w}_k)]\}, \quad (6)$$

$$u_k(x) = \operatorname{argmin}_{u \in U(x)} \{g_k(x, u, \hat{w}_k) + J_{k+1}[f_k(x, u, \hat{w}_k)]\}. \quad (7)$$

### B. Disturbance Sequences

In many cases, the generalized disturbance may be considered as the output of a very complicated stochastic system. For example, the driver's power demand is a reaction to the changing road profiles, traffic conditions, traffic lights, and pedestrians. In these cases, it is desired to develop a causal optimal controller with respect to not just the nominal disturbance, but all possible disturbance scenarios in the future. In other words, we want to make a prediction that includes all possible future scenarios with their probabilities, and find the optimal control with respect to this stochastic future.

We define a disturbance sequence as  $\mathbf{w} = [w_0, \dots, w_{N-1}] \in \mathcal{W}$ , where  $\mathcal{W}$  is the set of all possible disturbance sequences. We use  $\mathbf{w}(k)$  to denote the  $k$ -th element of this disturbance sequence, that is,  $w_k$ . For each possible disturbance sequence  $\mathbf{w}_i$ , its a priori probability is  $p(\mathbf{w}_i)$ . It is assumed that all possible disturbance sequences  $\mathcal{W}$  and the probabilities  $p(\mathbf{w}_i)$  of each  $\mathbf{w}_i \subseteq \mathcal{W}$  are predicted at step 0. At a future step  $k$ , no additional information about  $\mathbf{w}$  and the probabilities of  $\mathbf{w}$  is provided except the value of  $w_k$ . We use  $\mathcal{W}_a$  to denote the set containing all disturbance sequences with non-zero probabilities at step 0.

## III. OPTIMAL CONTROL WITH SCENARIO TREES

### A. Scenario Trees

We can use a tree structure to represent all disturbance sequences in the set  $\mathcal{W}_a$ . Suppose at step  $k$ , the disturbance sub-sequence that has been observed from step 0 to  $k$  is  $[w_0, w_1, \dots, w_k]$ . The information from this sub-sequence can be interpreted as that the full disturbance sequence  $\mathbf{w}$  must satisfy  $\mathbf{w}(0) = w_0, \dots, \mathbf{w}(k) = w_k$ . If we define a set function  $I(\cdot)$  on disturbance sub-sequences

$$\begin{aligned} &I_k([w_0, \dots, w_k]) \\ &= \{\mathbf{w} \in \mathcal{W}_a | \mathbf{w}(0) = w_0, \dots, \mathbf{w}(k) = w_k\}. \end{aligned} \quad (8)$$

then given disturbance observation  $[w_0, w_1, \dots, w_k]$ , we can define a set  $\mathcal{N}_k$  as  $\mathcal{N}_k = I_k([w_0, \dots, w_k])$  to represent the information we have learned from the disturbance history. We call  $I_k([w_0, \dots, w_k])$  the admissible disturbance sequence set at step  $k$ .

At every step  $k$ , based on the observation of the current and historical values of  $w$ , we can determine  $\mathcal{N}_k$ . It is not difficult to see that  $\mathcal{N}_k$  satisfies

$$\mathcal{W}_a \supseteq \mathcal{N}_0 \supseteq \mathcal{N}_1 \supseteq \dots \supseteq \mathcal{N}_{N-1} = \{\mathbf{w}\}, \quad (9)$$

where  $\mathbf{w}$  is the full disturbance sequence observed at step  $N - 1$ .

The stochastic dynamic evolution of the disturbance sequence set  $\mathcal{N}$  from step 0 to  $N - 1$  can be obtained when an initial value of  $\mathcal{N}$  is given. Before step 0, the only information about future disturbance is that  $\mathbf{w} \in \mathcal{W}_a$ , which means  $\mathcal{W}_a$  is the initial condition for  $\mathcal{N}$ . Due to the property shown in (9), it is intuitive to use a tree structure to represent these dynamics. This can be called a scenario tree.

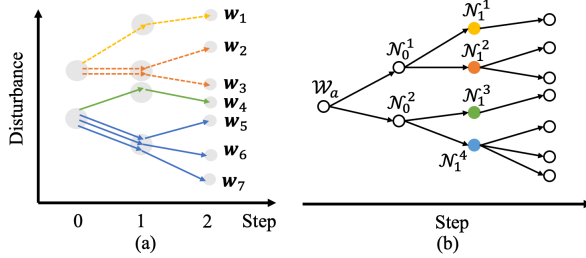


Fig. 1. A scenario tree example: (a) 7 disturbance sequences; (b) the scenario tree representation of the 7 disturbance sequences.

Fig. 1 provides an example of a scenario tree. There are 2, 4, and 7 possible  $w$  values at step 0, 1, and 2, respectively, and there are 7 possible disturbance sequences. In Fig. 1 (b), each node of the tree is essentially a set of disturbance sequences  $\mathcal{N}_k^i \subseteq \mathcal{W}_a$ . The root node, which is the initial set  $\mathcal{W}_a$ , contains all 7 sequences.  $\mathcal{N}_0^1$  contains the 3 dash-line sequences and  $\mathcal{N}_0^2$  contains the 4 solid-line sequences. At step 1, the four colored nodes  $\mathcal{N}_1^i$  contain the disturbance sequences of the corresponding color. At step  $k$ , the node  $\mathcal{N}_k$  can be considered as a measured state of the system, as  $w_0, \dots, w_k$  are known at step  $k$ . The arrows in Fig. 1 (b) can be considered as the possible state transitions of the tree node  $\mathcal{N}$ .

Define  $p(\mathcal{N}_k)$  as the probability of the node  $\mathcal{N}_k$ , which is the sum of the probabilities of all disturbance sequences in this set,

$$p(\mathcal{N}_k) = \sum_{\mathbf{w}_i \in \mathcal{N}_k} p(\mathbf{w}_i). \quad (10)$$

A node  $\mathcal{N}_k$  may have multiple branches leading to multiple child nodes at step  $k + 1$ . The conditional probability of the node  $\mathcal{N}_k$  moving to its child node

$\mathcal{N}_{k+1}$  can be obtained using the probabilities of the disturbance sequences in these two node sets,

$$\begin{aligned} P[\mathcal{N}_{k+1} = I_{k+1}(w_0, \dots, w_k, w_{k+1}) | \mathcal{N}_k = I_k(w_0, \dots, w_k)] \\ = p(I_{k+1}(w_0, \dots, w_k, w_{k+1})) / p(I_k(w_0, \dots, w_k)) \end{aligned} \quad (11)$$

In an exact scenario tree,  $\mathcal{N}_k$  has stochastic dynamics in the form of a Markov chain as follows,

$$P(\mathcal{N}_{k+1}^j | \mathcal{N}_k^i) = \begin{cases} p(\mathcal{N}_{k+1}^j) / p(\mathcal{N}_k^i), & \text{if } \mathcal{N}_{k+1}^j \subseteq \mathcal{N}_k^i, \\ 0, & \text{otherwise.} \end{cases} \quad (12)$$

## B. Integration in Optimal Control

Considering the system dynamics,

$$x_{k+1} = f_{\mathcal{N},k}(x_k, u_k, \mathcal{N}_k) = f_k[x_k, u_k, w_k(\mathcal{N}_k)], \quad (13)$$

where  $w_k(\mathcal{N}_k) = \mathbf{w}(k)$ ,  $\forall \mathbf{w} \in \mathcal{N}_k$ . As  $w_0, \dots, w_k$  is known,  $\mathcal{N}_k$  can be viewed as a measured state of the system. The feedback control policies  $\pi_k$  will be,

$$u_k = \pi_k(x_0, \dots, x_k, w_0, \dots, w_k) = \pi_k(x_k, \mathcal{N}_k). \quad (14)$$

Using the pair  $(x, \mathcal{N})$  as the new state, the predictive optimal control policy  $\pi^*$  can be obtained via dynamic programming backward induction.

One disadvantage of this scenario tree method is the exponential growing computational consumption with the number of steps and branches of the tree. In practice, it is common that a large number of possible future disturbance sequences are predicted, that is,  $|\mathcal{W}_a|$  can be indeed very large. To reduce the optimal policy computation load for a large number of disturbance sequences, we propose an approximation method for the scenario tree in the next section.

## IV. OPTIMAL CONTROL WITH SCENARIO TREE APPROXIMATION

### A. Scenario Tree Approximation

In this section, we propose a clustering-based approximation method for the scenario tree. Instead of determining the tree node  $\mathcal{N}_k$  using  $I_k([w(0), \dots, w(k)])$ , a different way to map a disturbance sub-sequence  $[w_0, w_1, \dots, w_k]$  to a node is used.

At each step  $k$ , a clustering function  $C_k$  is used to cluster disturbance sequences  $\mathbf{w}$  with similar sub-sequences  $[w(0), \dots, w(k)]$ . Clustering means classifying samples in the data set to a smaller number of clusters following specific rules that measure the similarity. Given any disturbance sub-sequence  $[w_0, \dots, w_k]$ ,  $C_k$  maps this sequence to an integer class number  $i$ ,  $i \in 1, 2, \dots, n_k$ , where  $n_k$  is the total number of clusters at step  $k$ .

Then in the scenario tree approximation, elements in one node is defined by the following set,

$$\mathcal{A}_k(i) = \{\mathbf{w} \in \mathcal{W}_a | C_k([\mathbf{w}(0), \dots, \mathbf{w}(k)]) = i\}. \quad (15)$$

All disturbance sequences in  $\mathcal{N}_k^i = \mathcal{A}_k(i)$  have their sub-sequences before step  $k$  belonging to a same cluster under  $C_k$ .

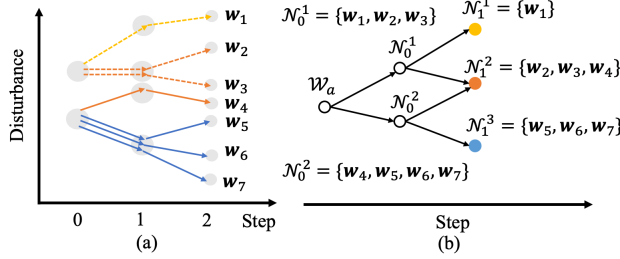


Fig. 2. A clustering-based scenario tree approximation example: (a) 7 disturbance sequences; (b) the 3-cluster approximation of the scenario tree at step 2

An example that illustrates the clustering of the scenario tree is shown in Fig. 2. We assume  $n_k = 3$ , that is, we want to use no more than 3 clusters at each step to approximate the tree structure. The clustering at step 0 is the same as the exact scenario tree, as the exact tree has only 2 nodes at step 0. At step 1, a clustering based on  $[\mathbf{w}(0), \mathbf{w}(1)]$  is performed, which generates 3 clusters:  $\mathcal{N}_1^1$ ,  $\mathcal{N}_1^2$ , and  $\mathcal{N}_1^3$ . After the clustering, the property in (9) no longer holds. Therefore  $\mathcal{N}$  is no longer in a tree structure as one child node  $\mathcal{N}_{k+1}$  may have multiple parent nodes  $\mathcal{N}_k$ . However,  $\mathcal{N}_k$  is still a measured state and its stochastic state transition is an approximation of the exact scenario tree.

In the scenario tree approximation,  $\mathcal{N}_k$  has the stochastic dynamics as follows,

$$P(\mathcal{N}_{k+1}^j | \mathcal{N}_k^i) = p(\mathcal{N}_{k+1}^j \cap \mathcal{N}_k^i) / p(\mathcal{N}_k^i) \quad (16)$$

The clustering-based approximation cuts down the number of states that representing disturbance from the number of all possible sub-sequences  $[w_0, \dots, w_k]$  to a preset number  $n_k$ . This significantly reduces the state space size for backward induction in dynamic programming. As the number of clusters  $n_k$  increases, the approximation becomes closer to the exact scenario tree. If the total sub-sequences number is large, we can even further reduce the computational load by random sampling in  $\mathcal{A}_k(i)$  to approximate the expected cost and transition probabilities for  $\mathcal{A}_k(i)$ .

### B. A Two-Level Control Architecture

One problem after the clustering approximation is that,  $\mathbf{w}(k)$  may take different values in the same  $\mathcal{N}_k$ .

This means that at step  $k$  the selection of a common control input  $u_k$  that works for all  $\mathbf{w} \in \mathcal{N}_k$  may be limited. In an extreme case, such a common  $u_k$  may not exist. We design an extra mapping layer to guarantee that the control policy can offer admissible control input to the system. An admissible control input  $u_k$  should guarantee that  $x_{k+1} \in X$ , where  $x_{k+1} = f_k(x_k, u_k, w_k)$ . Instead of finding a feedback control policy  $u_k = \pi_k(x_k, \mathcal{N}_k)$ , we find a higher-level control

$$v_k = \pi_k(x_k, \mathcal{N}_k), \quad (17)$$

and determine the lower-level control by

$$u_k = G_u(x_k, v_k, w_k) = \underset{\text{admissible } u_k}{\operatorname{argmin}} (||v_k - u_k||). \quad (18)$$

Essentially the feedback control policy becomes  $u_k = G_u[x_k, \pi_k(x_k, \mathcal{N}_k), w_k] = \pi'_k(x_k, \mathcal{N}_k, w_k)$ . Because  $G_u(\cdot)$  only depends on the values of  $x_k, v_k, w_k$ , we can define the mapping  $G_u(\cdot)$  before the design of optimal control, and consider it as part of the stochastic system dynamics.

### C. Integration in Optimal Control

In the cluster approximation, given  $\mathcal{N}_k$ ,  $w(\mathcal{N}_k) = \mathbf{w}(k)$ ,  $\mathbf{w} \in \mathcal{N}_k$  is a random variable. Therefore the system dynamics equation

$$\begin{aligned} f_{\mathcal{N},k}(x_k, u_k, \mathcal{N}_k) &= f_k[x_k, u_k, w_k(\mathcal{N}_k)] \\ &= f_k[x_k, G_u(x_k, v_k, w_k), w_k(\mathcal{N}_k)] \\ &= f_{\text{cluster},k}(x_k, v_k, \mathcal{N}_k) \end{aligned} \quad (19)$$

is a stochastic function whose output  $x_{k+1}$  is a random variable. The probability of  $x_{k+1}$  and  $\mathcal{N}_{k+1}$  can be determined if  $x_k, \mathcal{N}_k$  and  $v_k$  is given. The process of backward induction is similar to the exact scenario tree case, where the pair  $(x, \mathcal{N})$  is considered as the state of a stochastic system.

## V. NUMERICAL EXAMPLES AND RESULTS

In this section, we use two numerical examples to show the performance of the exact tree scenario and clustering-based approximation of the scenario tree.

We consider the system,

$$x_{k+1} = 0.9x_k + u_k - w_k, \quad (20)$$

where  $x, u$  and  $w$  are scalars. The system has a quadratic running cost

$$g_k(x_k, u_k, w_k) = u_k^2, \quad (21)$$

and a terminal cost

$$g_N(x_N) = (-2 - x_N)^2. \quad (22)$$

The initial condition is  $x_0 = 1.8$ , and  $N = 10$ . The constraints are  $x \in [-2, 2]$  and  $u \in [0, 1.6]$ . In dynamic programming,  $x$  and  $u$  are quantized with an interval of 0.02. Without loss of generality, we assume that every disturbance sequence in the predicted disturbance sequence set  $\mathcal{W}_a$  has equal probabilities.

#### A. A Simple Scenario Tree

A simple  $\mathcal{W}_a$  with 4 disturbance sequences as shown in Fig. 3 is considered. All sequences converge to a single point at step 4 and 5. We use this example to show why in some cases a scenario tree representation may be more appropriate than a non-stationary Markov chain representation. The transition probability matrix  $T_k$  of this non-stationary Markov chain at step  $k$  can be defined as

$$T_k(i, j) = P[w_{k+1} = w_{k+1}^j | w_k = w_k^i], \quad (23)$$

for every possible  $w_k^i$  and  $w_{k+1}^j$ .  $T_k$  is computed using the disturbance sequence data for  $k = 0, 1, \dots, N - 1$ .

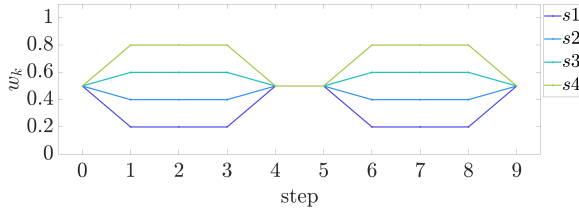


Fig. 3. A simple disturbance scenario of 4 sequences. The 4 disturbance sequences are defined as s1, s2, s3, and s4.

We use dynamic programming to compute two policies with two methods: (1) using the exact scenario tree to represent the future disturbance sequences and (2) using the non-stationary Markov chain to represent the future disturbance sequences.

The average cost for four disturbance sequences under the two methods is shown in Table I. In this simple tree structure, the limitation of the Markov chain can be seen. The performance of the optimal predictive control with the scenario tree representation is better than that using the Markov chain. This is due to the fact that the scenario tree keeps the multi-step long-term information of the disturbance sequences, while the Markov chain representation loses this information and only considers transitions between two steps.

#### B. A Complex Scenario Tree with Approximation

In this example, we use a more complex scenario tree to demonstrate the performance of the clustering-based tree approximation method. 1447 disturbance sequences are generated as shown in Fig. 4.

TABLE I  
AVERAGE COST RESULTS OF THE 4 DISTURBANCE SEQUENCES

Method	Average cost	Computation time
Exact Scenario Tree	0.3180	5.52s
Non-Stationary Markov Chain	0.3429	18.29s

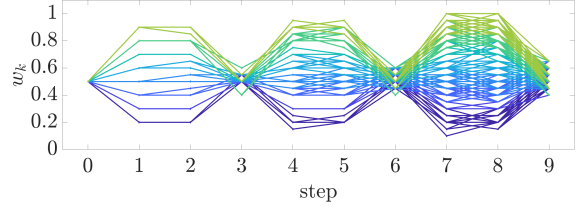


Fig. 4. A complex disturbance scenario of 1447 sequences.

We use dynamic programming to compute two policies with three methods: (1) using the exact scenario tree representation, (2) using the clustering-based scenario tree approximation representation, where  $n_k = 10$  (10 clusters for each step), and (3) using the non-stationary Markov chain to represent the future disturbance sequences. In this example, we use the k-means clustering [14] as the function  $C_k$  to cluster the disturbance subsequences. For k-means, we wish to choose  $n_k$  centers  $\mathcal{C}$  of the data to minimize the potential function,

$$\phi = \sum_{w \in W} \min_{c \in \mathcal{C}} \|w - c\|^2, \quad (24)$$

The clustering result examples at step 6 and 9 are shown in Fig. 5. In the non-stationary Markov chain method, to keep the size of the state space the same as the clustering-based method, we quantize the disturbance to a 10-value grid and obtain a time-dependent transition matrix using (23).

TABLE II  
AVERAGE COST RESULTS OF THE 1447 DISTURBANCE SEQUENCES

Method	Average cost	Computation time
Exact Scenario Tree	0.4407	487.90s
Clustering-Based Approximation	0.4432	32.52s
Non-Stationary Markov Chain	0.4992	17.67s

The average cost for all 1447 disturbance sequences is shown in Table II and the state trajectories are shown in Fig. 6. For the computation time, all results are from codes written in MATLAB and run on a Mac laptop with Intel Core i7-9750H. To optimize computing efficiency for the clustering-based method, we build a static table that maps the discretized  $x_k, u_k, w_k$  values to corresponding  $x_{k+1}$  and  $l_k$ . It takes 6.3s to build

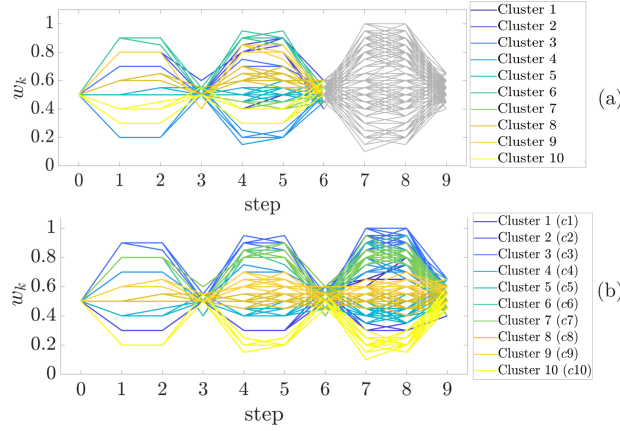


Fig. 5. Clustering results for disturbance sequences at (a) step 6, and (b) step 9.

the static map (included in the 32.52s computation time). The map can be reused for updating policies from different clustering approximation results unless the system dynamics or the cost function changes. The clustering-based method performs much better than the non-stationary Markov chain method and ensures a similar performance as the exact scenario tree method. However, the clustering approximation method significantly reduces computation time from the exact tree method. The clustering-based method captures more disturbance sequence features from previous steps while the Markov chain loses this information.

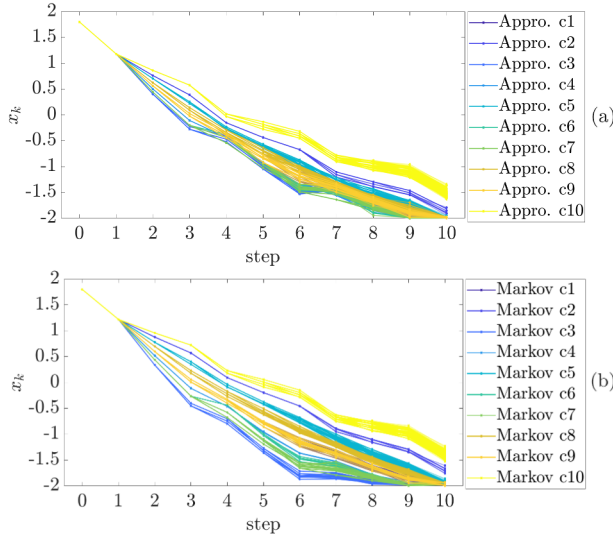


Fig. 6. State trajectories for (a) the clustering-based scenario tree approximation method and (b) the non-stationary Markov chain method of the 1447 disturbance sequences. Each colored trajectory is corresponding to a disturbance sequence in one cluster of step 9.

## VI. CONCLUSIONS

In this paper, a predictive optimal control method with a clustering-based scenario tree approximation of disturbance sequences is proposed. The method is demonstrated in numerical examples, where the proposed method performs better than the traditional method which considers the disturbance as a Markov chain.

## REFERENCES

- [1] X. Zeng and J. Wang, "Worst-case relative cost optimal control for dynamic systems with finite admissible disturbance sequence sets," *International Journal of Control*, pp. 1–8, 2019.
- [2] G. Ripaccioli, D. Bernardini, S. Di Cairano, A. Bemporad, and I. Kolmanovsky, "A stochastic model predictive control approach for series hybrid electric vehicle power management," in *Proceedings of the 2010 American Control Conference*. IEEE, 2010, pp. 5844–5849.
- [3] C. Sun, X. Hu, S. J. Moura, and F. Sun, "Velocity predictors for predictive energy management in hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 3, pp. 1197–1204, 2014.
- [4] X. Zeng and J. Wang, "A parallel hybrid electric vehicle energy management strategy using stochastic model predictive control with road grade preview," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 6, pp. 2416–2423, 2015.
- [5] S. J. Moura, H. K. Fathy, D. S. Callaway, and J. L. Stein, "A stochastic optimal control approach for power management in plug-in hybrid electric vehicles," *IEEE Transactions on control systems technology*, vol. 19, no. 3, pp. 545–555, 2010.
- [6] D. de la Penad, A. Bemporad, and T. Alamo, "Stochastic programming applied to model predictive control," in *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005, pp. 1361–1366.
- [7] S. Di Cairano, D. Bernardini, A. Bemporad, and I. V. Kolmanovsky, "Stochastic mpc with learning for driver-predictive vehicle control and its application to hev energy management," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 3, pp. 1018–1031, 2013.
- [8] Q. Meng and T. Wang, "A scenario-based dynamic programming model for multi-period liner ship fleet planning," *Transportation Research Part E: Logistics and Transportation Review*, vol. 47, no. 4, pp. 401–413, 2011.
- [9] H. Borhan, A. Vahidi, A. M. Phillips, M. L. Kuang, I. V. Kolmanovsky, and S. Di Cairano, "Mpc-based energy management of a power-split hybrid electric vehicle," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 3, pp. 593–603, 2011.
- [10] D. Rotenberg, A. Vahidi, and I. Kolmanovsky, "Ultracapacitor assisted powertrains: Modeling, control, sizing, and the impact on fuel economy," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 3, pp. 576–589, 2010.
- [11] M. P. O’Keefe and T. Markel, "Dynamic programming applied to investigate energy management strategies for a plug-in hev," National Renewable Energy Lab.(NREL), Golden, CO (United States), Tech. Rep., 2006.
- [12] L. Johannesson, M. Asbogard, and B. Egardt, "Assessing the potential of predictive control for hybrid vehicle powertrains using stochastic dynamic programming," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 1, pp. 71–83, 2007.
- [13] D. P. Bertsekas, *Approximate dynamic programming*. Athena scientific Belmont, 2012.
- [14] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," Stanford, Tech. Rep., 2006.