Original Article

Institution of MECHANICAL ENGINEERS
Journal of AEROSPACE ENGINEERING

Proc IMechE Part G:
J Aerospace Engineering
2023, Vol. 237(10) 2240–2251

DOI: 10.1177/09544100221149231
journals.sagepub.com/home/pig
Sage

Co-design of active vibration control and optimal sensor and actuator placement for a flexible wing using reinforcement learning

Tianyi He[1] and **Weihua Su**[2]

Abstract

This paper presents applying reinforcement learning to find the optimal sensor/actuator placement (OSAP) policy and optimal control for the flexible wing. The "co-design" objective is to find the OSAP and its associate controller to render the optimal closed-loop performance. The nonlinear vibration dynamics of the flexible wing are modeled in the linear parameter varying (LPV) approach so that LPV-H_∞ controllers can be designed. The co-design problem is formulated into mixed-integer semi-definite programming (MISDP). As a special form of combinatorial optimization, MIDSP solves integer optimization for sensor/actuator selection and convex optimization for controller design. A modified reinforcement learning algorithm is applied to solve this NP-hard optimization problem and obtain a converged solution. In addition, RL is compared with the greedy algorithm and genetic algorithm to demonstrate its strengths and drawbacks in solving high-dimensional MISDP. The solutions obtained by RL and the greedy algorithm are verified and compared in the high-fidelity simulation with the full-order model.

Keywords

Co-design, structure control, sensor/actuator placement, flexible wing, reinforcement learning

Date received: 27 June 2022; accepted: 15 December 2022

Introduction

The flexible aircraft wing has been recognized as a promising design for high-altitude long-endurance (HALE) aircraft. Its flexible structure results in high aerodynamic performance and improved fuel efficiency but leads to reduced stability margin at the same time.[1–3] Under the same flight condition, the flutter is more severe than that of the traditional rigid wings. Therefore, active vibration control is essential for suppressing flutters to avoid structural failure and increase the flexible wing's flight stability margin.[4,5]

Early research revealed that the vibrations on a flexible wing are nonlinear and parameter-dependent on flight condition.[1] One challenge of designing a gust alleviation system is that sensor and actuator selections are not independent of the control design. The placements of sensors and actuators determine the controller complexity and the best achievable closed-loop performance. However, the sensor/actuator placement is designed traditionally by heuristic methods, which are rarely considered together with controller synthesis. Consequently, the control system will only lead to sub-optimal closed-loop performance, and the deviation from optimum is not guaranteed. Another challenge is the instability of the flexible wing. When flight speed is beyond a certain value, the vibration mode will change from stable to unstable. Therefore, a systematic method is needed to co-design the optimal sensor/actuator placement (OSAP) and the active controller to achieve optimal closed-loop performance. For the flexible wing, the co-design of OSAP and gain-scheduling control is still an open problem.

One of the most straightforward strategies is to place sensors and actuators to avoid right half-plane zeros.[6] The reason is that the unstable zeros will limit the bandwidth of control inputs and consequently limit the closed-loop performance.[7] Quantitative measures of the transfer function, for example, the singular value[8] and the condition number,[7] can be optimized. But optimizing these frequency-dependent functions is not efficient for large-

[1] Department of Mechanical and Aerospace Engineering, Utah State University, Logan, UT, USA
[2] Department of Aerospace Engineering and Mechanics, The University of Alabama, Tuscaloosa, AL, USA

Corresponding author:
Tianyi He, Department of Mechanical and Aerospace Engineering, Utah State University, 4130 Old Main Hill, Logan, UT 84322, USA.
Email: tianyi.he@usu.edu

scale input–output systems. For LTI systems in the state-space representation, quantitative measures, often linked with time-domain physical meanings, can be formulated and optimized more easily. The optimization-based methods in the literature can be categorized by objectives and searching methods.

The most common quantitative measure for open-loop stable LTI systems is related to the state controllability Gramian (W_c) and observability Gramian (W_o). The sensor and actuators are selected to maximize the minimum eigenvalue of W_c and W_o, such that the input energy is minimized and output energy is maximized.[9,10] Other measures include determinants of W_c and W_o and reciprocals of $trace(W_c^{-1})$ and $trace(W_o^{-1})$.[11,12] These Gramian measures require the system to be open-loop stable. However, it is a challenge to quickly solve the Gramians for a high-dimensional system and parameter (time)-varying system. One exception is when the damping coefficient is small enough such that the Gramian can be approximated to diagonal, and the dominating sensors/actuators can be easily selected.[13] When the system is subject to external disturbance, H_2 and H_∞ norms of the closed-loop system are adopted as objectives.[14,15] The effective independence (EFI) method[16] selects measurement locations that make the mode shapes as linearly independent as possible. This method is equivalent to the minimization of the condition number of the information matrix.[17]

The optimization-based approach treats the sensors and actuators as discrete variables, and the nature of OSAP is a combinatorial optimization (CO) problem.[18] The methods used to solve this NP-hard problem fall into four groups: (1) exact methods, (2) relaxation-based convex optimization, (3) heuristic methods, and (4) learning-based methods.

Exact optimal solutions can be obtained by enumerating or using branch-and-bound.[19] However, this approach is only limited to small-size problems. The relaxation-based methods find an approximate solution by relaxing optimization in discrete space to continuous space. For instance, the binary variable $\{0, 1\}$ can be relaxed to continuous variable $[0, 1]$, and the final solution is the nearest integer rounded from the solution of convex optimization.[20] However, those approximate solutions deviate from the optimum, and the deviation error bound is not guaranteed.

Heuristic methods produce solutions that are not guaranteed to be optimal but in an efficient manner. If the objective functions are submodular, then the greedy algorithm can reach a near-global-optimal solution by taking local optimal steps.[21] However, the submodular property is a strict requirement and needs carefully checking of objective functions. Jawaid and Smith (2015)[21] gave some counterexamples that overturned proven submodular functions. The genetic algorithm (GA)[22] can solve OSAP effectively. These heuristic methods provide a practical and fast approach, but a heuristic naturally requires prior experience, and heuristic tricks influence the search algorithm's complexity and performance.

Learning-based methods, especially reinforcement learning (RL), are successfully applied to solve CO recently.[23,24] The learning-based methods actively learn and improve the heuristic in an iterative manner. The RL agent can automatically learn the optimal policy by interacting with an initially unknown environment, collecting the reward, and updating the policy. In Cappart et al. (2019),[25] the RL is used to solve the maximum independent set problem. Kasper et al. (2015)[26] proposes a machine learning–based algorithm of constrained sensor placement to recover a high-dimensional field from a finite number of local measurements with a linear estimator. Reinforcement learning is applied in the spatial domain for modeling distributed parameter systems, and the feasibility and efficiency are demonstrated by experimental results in Wang et al. (2019).[27] Those studies reveal the possibility of using the RL algorithm to tackle the co-design of OSAP and control for the flexible wing. The reduced requirements on the model make the RL application scope wider than these traditional methods in the literature.

In this paper, the co-design of OSAP and active control for the highly nonlinear flexible wing is solved by RL to achieve optimal vibration suppression against external disturbance. The placements of sensors and actuators are formulated to discrete optimization to select from a set of available sensors and actuators locations. The nonlinear and parameter-dependent properties of the vibration dynamics are described by a linear parameter-varying (LPV) model that is a polynomially parameter-dependent system. The H_∞ norm is used to evaluate the closed-loop performance by suppressing vibrations excited by external disturbance. The co-design problem is formulated as a mixed-integer semi-definite programming (MISDP), which is a combinatorial (hybrid) optimization problem with integer variables and real matrix-valued variables. Integer variables describe the sensor/actuator selection from a given set, and real matrix-valued variables determine the controller which achieves optimal H_∞ performance.

The CO problem is then solved using a modified RL to obtain the convergent solution. The convergent solution is given and verified for a flexible wing model. The RL-based method is then compared with the greedy algorithm (heuristic) to demonstrate the improved gust alleviation performance by the high-fidelity simulation results.

The main contributions of this paper are two-fold:

(1) Formulating the co-design problem of LPV control with input constraints and sensor/actuator placement, into a combinatorial optimization for the flexible wing. The gust alleviation of flexible wing is the objective to be optimized jointly by control variables and sensor/actuator integer variables.
(2) Using the reinforcement learning method to solve the mixed-integer semi-definite programming and verifying the optimal "co-design" in high-fidelity simulation. The combinatorial optimization consists of nonlinearity introduced by the parameter-dependent polynomials and discrete variables for sensor/actuator placement.

The rest of this paper is organized as follows. Firstly, the basics of linear parameter varying (LPV) modeling and LPV controller synthesis of the flexible wing are provided, and then, the co-design problem is formulated into a CO problem. After that, the RL algorithm is applied to the co-design problem, and then, optimization solutions and verification are shown to demonstrate the effectiveness. The comparison results of RL with other methods are then given and discussed. At last, conclusions are made. The following notations are used in this paper. $|S|$ denotes the cardinality of a set S, and $S \setminus s_k$ denotes removing an element s_k from S. The vertices of a polytope Θ are denoted by $ver(\Theta)$. A positive definite matrix M is written as $M > 0$ and $\langle M \rangle = M + M^T$.

Problem formulation

LPV model of nonlinear aeroelasticity of a flexible wing

Figure 1 presents the schematic diagram of the blended-wing-body (BWB) aircraft with flexible wings. The modeling and control of the flexible wing have been reported in previous work. Su and Cesnik (2010)[1] developed the nonlinear aeroelastic model of the flexible wing, Jiboory et al. (2017)[28] derived a reduced-order LPV model, and He et al. (2017)[29,30] developed the LPV controllers. The LPV modeling process mainly consists of four steps. In the first step, the wing is gridded into a finite number of beam elements, and the nonlinear aeroelastic model is developed. The second step is to linearize the nonlinear model at gridded flight speeds and conduct coordinate transformation to the modal form. As a result, a series of LTI full-order models (FOMs) are obtained. The third step is to align the vibration modes to depict the evolution of vibration modes so that model reduction can be performed by keeping the most significant modes.[28] In the last step, the evolution of modal LTI models is interpolated into a polynomial parameter-dependent LPV model.

Each wing of the BWB aircraft is gridded into 12 beam elements, and the inner nine elements at each wing are locations where control surfaces can be placed, labeled as $U1 - U9$. The vertical displacements at the equally spaced locations along the wingspan are selected as performance outputs that reflect the vibration behaviors of the entire wing. The open-loop nonlinear and linearized flight dynamic and aeroelastic models were created by following the work,[1] where the nonlinear aeroelastic formulation was developed based on a strain-based geometrically nonlinear beam theory.[31] The beam theory is geometrically exact, considering the composite cross-sectional properties that can be obtained using the variational asymptotic approach.[32] Due to the symmetry, only the right-side wing is considered.

The aircraft is assumed to cruise at 10,000 feet with a flight speed θ from 80 to 130 m/s. At gridded flight speeds with an increment of 0.5 m/s, the nonlinear structure vibration behaviors are analyzed in the modal coordinate. The system matrices at the gridded point take the form of modal coordinates. Each of the mode consists of a pair of conjugate eigenvalues $\alpha_i \pm j\beta_i$, $i = 1, 2, \ldots, 6$. Only six dominant modes remained in the reduced-order model, marked as M1-M6, and the physical meaning of each mode is explained in Table 1. The eigenvalue solution yields coupled modes, where each mode consists of elastic, flight dynamic (rigid body), and aerodynamic components. It's worth mentioning that, from the analysis of obtained eigenvectors, there does not exist the mode of first anti-symmetric out-of-plane bending of the wing coupled with rigid-body roll. However, the modeling approach does not exclude such a fundamental mode if it exists in other different aircraft models.

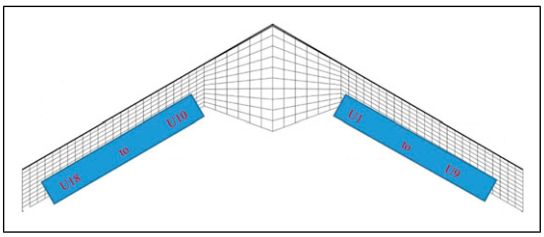

Figure 1. Schematic layout of BWB airplane configuration.

The evolution of each mode relative to varying flight speed is plotted by the gray triangles in Figure 2. At a flight speed of 115 m/s, the mode M1 becomes unstable. The evolutionary trajectories of these vibration modes clearly show the nonlinear dependency on varying flight speeds. Thus, a polynomial LPV model is used to describe the nonlinear dependence on the varying parameter (flight speed).

Considering the scenario that the flexible wing is perturbed by gust disturbance and measurement noises, the LPV model in (1) describes the perturbed system

$$\begin{aligned} \dot{x}_p(t) &= A(\theta)x_p(t) + B_w(\theta)w(t) + B_2(\theta)u(t) \\ z(t) &= C_1(\theta)x_p(t) + D_{12}(\theta)u(t) \\ y(t) &= C_y(\theta)x_p(t) + v(t) \end{aligned} \tag{1}$$

where $\theta(t)$ denotes the scheduling parameter, that is, flight speed, $x_p(t)$ denotes the model state, and $u(t)$ are the deflection angles of control surfaces installed at selected locations; $z(t)$ are the controlled outputs that consist of weighted control inputs and vertical bending displacements at all locations; and $y(t)$ are the measurements from installed sensors at selected locations. The measurement noise $v(t)$ is assumed to be zero-mean, Gaussian white noise, $E\{v(s)v^T(t)\} = V\delta(t-s)$, and $w(t)$ is the external disturbance exciting the vibrations, for example, the gust disturbance.

Because the nonlinear dependency on flight speed is approximated to polynomial, the system matrices are in the second-order polynomial parameter-dependent form in (2). The LPV matrices $A(\theta)$ are explained in Appendix and are available to download in the url link[1]

$$A(\theta(t)) = A_0 + A_1\theta(t) + A_2\theta^2(t). \tag{2}$$

Table 1. Mode description in the reduced-order model.

Mode ID	Rigid body	Flexible	Note
M1	Plunge and pitch	1st symmetric out-of-plane bending	Bending/torsion coupling
M2	Plunge and pitch	2nd symmetric out-of-plane bending	Bending/torsion coupling
M3	Plunge and pitch	1st symmetric in-plane bending	Bending/torsion coupling
M4	Roll	2nd anti-symmetric out-of-plane bending	Bending/torsion coupling
M5	—	1st anti-symmetric in-plane bending	Bending/torsion coupling
M6	—	—	Aerodynamic dominant mode

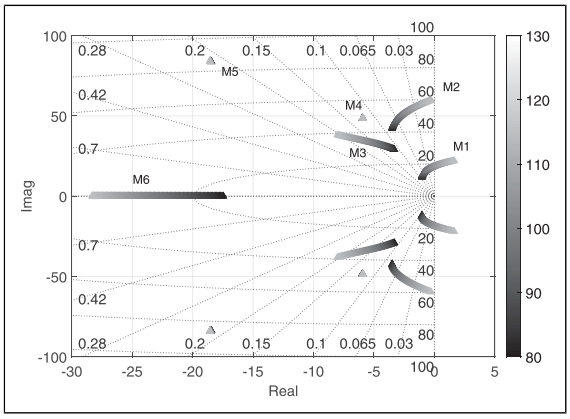

Figure 2. Root loci of open-loop vibration modes with varying flight speed, color bar: 80–130 m/s.

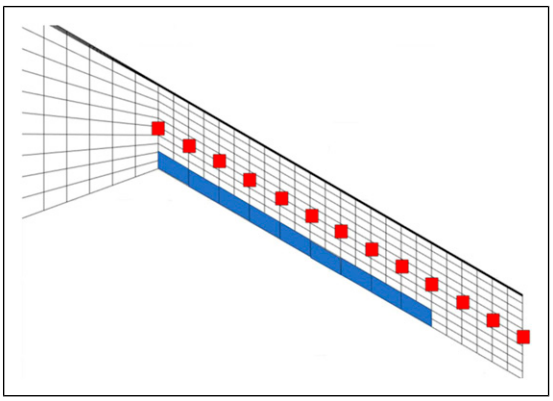

Figure 3. Sensor and actuator candidate locations, red: sensors candidate locations labeled from left to right as y_1 to y_{13}, blue: actuators (flaps) candidate locations, labeled from left to right as U_1 to U_9.

The bounds of pair $(\theta, \dot{\theta})$ are represented as

$$\begin{aligned} \theta &\in \Theta = \left\{\underline{\theta} \le \theta(t) \le \overline{\theta}\right\}, \\ \dot{\theta} &\in \Lambda = \left\{-v_\theta \le \dot{\theta}(t) \le v_\theta\right\}. \end{aligned} \tag{3}$$

There are two benefits resulting from using the polynomial parameter-dependent form. Firstly, the polynomial function can more accurately capture the nonlinear dependency on varying speeds rather than linear polytopic representation. The matrices A_0, A_1, and A_2 are easy to be computed from least-square regression. Secondly, the polynomial parameter dependency renders LMI characterizations of controller synthesis over polynomials. Available optimization tools can solve LMIs efficiently to synthesize the controllers.[33]

Sensor and actuator selection by projection

As shown in Figure 3, the sensor location candidates are marked by red squares, and they are equally spaced on the wingspan. The actuator location candidates are marked by blue parallelograms. $\overline{y}(t)$ denote measurement signals from all candidates $\overline{y}(t) = [\overline{y}_1(t), \overline{y}_2(t), \ldots, \overline{y}_{m_1}(t), \ldots, \overline{y}_{M_1}(t)]$, where M_1 is the total number of sensor location candidates. $N_1 \le M_1$ number of the sensor locations are selected to install sensors to obtain the actual measurement $y(t)$. In practice, the sensors and actuators are often installed symmetrically, so the symmetric selection is adopted for both sides. Since there are 12 gridded elements in total on the right wing, it is assumed that the available measurement locations are the edge points indicated by the red square. The total number of sensor placements is 13. The inner 9 elements are assumed to be the candidate locations for the flaps (control surfaces), so the total number of actuators' installation places is 9.

The projection operation produces sensor selection from the entire set of available measurements to the selected measurement subset. The projection operator is denoted by $P_y : R^{M_1} \mapsto R^{N_1}$, leading to $y = P_y \overline{y}$.

Each of the sensor locations is represented by an integer. Then, selecting N_1 number of sensors is equivalent to forming the set S_y by selecting N_1 number of integer variables from the set $S\overline{y} = \{1, 2, \ldots, M_1\}$.

Similarly, selecting $N_2 \le M_2$ number of actuator candidates from the entire set of M_2 locations is realized by the projection operator $P_u : R^{M_2} \mapsto R^{N_2}$, which leads to $u = P_u \overline{u}$. Then, selecting N_2 number of sensors is mathematically equivalent to forming the set S_u by selecting N_2 number of integer variables from the set $S_{\overline{u}} = \{1, 2, \ldots, M_2\}$.

For example, selecting u_1 and u_2 from the set $\{u_1, u_2, u_3\}$ is realized by the projection, where the corresponding element of selected integer variable is 1, and unselected are 0. Therefore, $\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}}_{P_u} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$.

The input matrix B is projected from the initial input matrix $\overline{B}_2$ by the projection matrix P_u^T

$$\underbrace{\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}}_{B_2} = \underbrace{\begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \end{bmatrix}}_{\overline{B}_2} \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}}_{P_u^T} \tag{4}$$

The bending displacements at all equally spaced locations are put in the performance output $z(t)$ so that the overall wingspan's vibrations are considered. Besides, the weighted control inputs by $W_{\overline{u}}$ are included to avoid the singular problem. The disturbance perturbs the system by $D_{12} = \overline{D}_{12} P_u^T, D_{21} = P_y \overline{D}_{21}$. Therefore, the system with the projection operator is expressed in equation (5), where the parameter dependency is omitted

$$\begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{bmatrix} = \begin{bmatrix} A & [B_w \ \ 0] & \overline{B}_2 P_u^T \\ \begin{bmatrix} C_{\overline{y}} \\ 0 \end{bmatrix} & 0 & \begin{bmatrix} 0 \\ W_{\overline{u}} P_u^T \end{bmatrix} \\ P_y C_{\overline{y}} & P_y [0 \ \ V] & 0 \end{bmatrix} \tag{5}$$

Using the selected sensor measurements and control surfaces, the dynamic output-feedback (DOF) LPV controller $K(\theta)$ in (6) is to be designed

$$K(\theta): \begin{cases} \dot{x}_K = A_K(\theta) x_K + B_K(\theta) y \\ u = C_K(\theta) x_K \end{cases} \tag{6}$$

The closed-loop system matrices are thus derived as

$$\begin{bmatrix} A_{cl} & B_{cl} \\ C_{cl} & D_{cl} \end{bmatrix} = \begin{bmatrix} A & \overline{B}_2 P_u^T C_K & B_1 \\ B_k P_y C_{\overline{y}} & A_K & B_K D_{21} \\ C_{\overline{y}} & 0 & 0 \\ 0 & W_{\overline{u}} P_u^T C_k & 0 \end{bmatrix} \tag{7}$$

LPV controller synthesis conditions

The H_∞ performance is to assess the closed-loop system robustness in suppressing external disturbance. Mathematically, let T_∞ denote the transfer function from external disturbance $w(t)$ to performance output $z(t)$, and $\|T_\infty\|_\infty$ is the H_∞ norm of T_∞. Then, the H_∞ performance for ($w(t)$, $z(t)$) pair is defined as

$$\|T_\infty\|_\infty = \sup_{w, z \in l_2, \|w\|_2 \neq 0} \frac{\|z(t)\|_2}{\|w(t)\|_2}. \tag{8}$$

With projections P_u and P_y, the following Theorem 1 provides the controller synthesis conditions for the H_∞ LPV DOF controller.

Theorem 1. *For the LPV system* (5)*, the gain-scheduling controller* (12) *minimizes the closed-loop H_∞ performance index in* (9)

$$\min_{X, Y, \widehat{A}_K, \widehat{B}_K, \widehat{C}_K, \widehat{D}_K} \gamma \tag{9}$$

if there exist parameter-dependent positive definite matrices X and Y, and parameter-dependent controller variables $\widehat{A}_K$, $\widehat{B}_K$, and $\widehat{C}_K$ such that (10)–(11) *hold for any $(\theta, \dot{\theta}) \in ver(\Theta \times \Lambda)$. The gain-scheduling controllercan be reconstructed as* (12)

$$\begin{bmatrix} -\dot{X} + \langle AX + \overline{B}_2 P_u^T \widehat{C}_K \rangle & * & * & * \\ \widehat{A}_K + A^T & \dot{Y} + \langle YA + \widehat{B}_K P_y C_{\overline{y}} \rangle & * & * \\ B_1^T & \left(YB_1 + \widehat{B}_K P_y \overline{D}_{21}\right)^T & -\gamma I & * \\ C_1 X + \overline{D}_{12} P_u^T \widehat{C}_K & C_1 & D_{11} & -\gamma I \end{bmatrix} < 0 \tag{10}$$

$$\begin{bmatrix} \overline{U}_k & e_k \widehat{C}_K & 0 \\ * & X & I \\ * & I & Y \end{bmatrix} > 0, \ k = 1, 2, \ldots, N_2. \tag{11}$$

$$\begin{cases} A_K = (N)^{-1} \left[\widehat{A}_K - YAX - \widehat{B}_K C_y X - YB_2 \widehat{C}_K \right] (M)^{-T} \\ B_K = N^{-1} \widehat{B}_K \\ C_K = \widehat{C}_K (M)^{-T} \end{cases} \tag{12}$$

Remark 1. *The proof is omitted and the detailed proof can be found in references.*[34,29] *In the controller reconstruction, the $\dot{\theta}$ dependency in the controller can be eliminated, with introduced conservativeness, by setting either X or Y as a constant matrix.*[34] *In addition, the variables $\widehat{A}_K$, $\widehat{B}_K$, $\widehat{C}_K$, $\widehat{D}_K$, and $Y$are also assumed in the second-order polynomial parameter-dependent form. For example, $\widehat{A}_K(\theta)$ is expressed as $\widehat{A}_K(\theta) = \widehat{A}_{K0} + \widehat{A}_{K1}\theta + \widehat{A}_{K1}\theta^2$. Therefore,*

variables $(\widehat{A}_{Ki}, \widehat{B}_{Ki}, \widehat{C}_{Ki}, Y_i, X_0)$, $i = 0, 1, 2$, *are to be sought to determine the controller that minimizes the closed-loop performance. This semi-definite program (SDP) is a convex optimization problem and it can be effectively solved using the parser ROLMIP,*[33] *YALMIP,*[35] *and the solver SEDUMI.*[36]

Remark 2. *One can easily transform the set of integer variables to the set of binary variables. For example, selecting integer* $\{1, 2\}$ *from the set* $\{1, 2, 3\}$ *is same as setting binary variables* $s_1 = 1$, $s_2 = 1$, $s_3 = 0$ *in the set* $\{s_1, s_2, s_3\}$. *Therefore, the mixed-integer SDP can be transformed into a mixed-binary SDP.*

Co-design as a combinatorial optimization

In the co-design of optimal sensor and actuator placement and optimal control, the sensor/actuator set and controller matrices are simultaneously sought to minimize the closed-loop performance. Therefore, the co-design problem is formulated as an optimization problem (13)

$$\min_{\substack{|S_y|=N_1,\,|S_u|=N_2 \\ S_y \subset S_{\bar{y}},\, S_u \subset S_{\bar{u}}}} \quad \min_{\left(\hat{A}_K, \hat{B}_K, \hat{C}_K, \hat{D}_K, Y, X\right)} \gamma \tag{13}$$

subject to LMIs (10) and (11) for $(\theta, \dot{\theta}) \in ver(\Theta \times \Lambda)$.

The optimization problem is now formulated as MISDP, a special type of CO. This CO problem is well-known as an NP-hard problem, which involves integer optimization for sensor/actuator selection and convex optimization for controller synthesis. The optimization complexity increases exponentially with the number of available sensors and actuators. The traditional searching algorithm is either inefficient or leads to a loss of optimality. Firstly, this complex CO cannot be directly solved by the traditional methods, like the gradient method, unless the discrete variables are relaxed to be continuous. The optimality may be sacrificed due to the relaxation. Secondly, even after relaxation, the resulting co-design optimization is still a non-convex optimization problem due to the coupling terms by sensor/actuator variables and controller variables. See LMI in (10), (11) for the non-convexity due to coupling between $(\widehat{A}_K, \widehat{B}_K, \widehat{C}_K)$ and (P_u, P_y). The optimization algorithms are often not efficient in solving high-dimensional non-convex problems. For example, the reduced-order model of the flexible wing has a dimension of 12, and it has 13 outputs and 9 inputs. Therefore, the LMI in Theorem 1 has 22 binary variables and 1380 continuous variables. The relaxation-based method will solve a non-convex problem with around 1400 variables and with nonlinear coupling between variables. The algorithm efficiency and global optimum are challenging to achieve. Therefore, the RL is used to tackle the resulting CO and search for a fast convergent solution.

Solving combinatorial optimization by reinforcement learning

Combinatorial optimization has experienced boosting progress from the success of RL and more broadly machine learning. The basic idea is iteratively learning useful heuristics.[23] The RL algorithm starts from an initial policy, learns from a policy's rewards, and improves the policy iteratively. Compared to the relaxation-based optimization, there is no introduced conservatism by relaxations. Compared to other stationary heuristic methods like the greedy algorithm, the RL has an adaptive heuristic strategy to obtain improved solutions over the static heuristic methods. This section will show how the RL technique is specifically applied to solve the co-design problem.

MDP of selecting sensors and actuators

RL is usually described as a Markov decision process (MDP). One sequence of MDP consists of basic elements (s, a, s', r), where s is the current state, a is the action, s' is the next state, and r is the immediate reward. States s, s' belong to the set of states $\mathbf{S}$. The action a belongs to the set of actions $\mathbf{A}$. The transition model $\mathbf{P}_a(s, s')$ determines the next state s' from the current state s under the action a. The reward model $\mathbf{R}_a(s, s')$ determines the immediate reward r after state transition from s to s' under the action a.

The search process for optimal sensor and actuator placement is a deterministic MDP. The state s represents the current sensor/actuator selection, equivalently, the set of selected integer variables. Starting from an initial state that selects all sensors and actuators, the action a is to remove one sensor/actuator from the selected set. s' is the new selected sensor/actuator locations. The future state s' is independent of history states and only depends on the current state and action. Therefore, this process satisfies the Markov property, and the state transition model $\mathbf{P}_a(s, s')$ is deterministic. The reward r is the optimal closed-loop performance using the selected sensors and actuators, which is *unknown* until executing the controller synthesis. Therefore, the reward model is not priorly known. The state value function $v(s)$ evaluates how good a state is, in terms of the best achievable closed-loop performance. The simple tabular method is used for the state value function. The iterations on the state value function carry out the search process for the optimal placement policy until convergence. With enough iterations of exploration, the global optimum can be achieved.[37] Because no relaxations or approximations are used, the optimality is not lost. However, it is worth mentioning that the tabular method needs a large space of states for the co-design problem due to the high-dimensional system. An alternative approach to implementing state value function is to use function approximations or neural network,[38] and different policy search methods can be used.

RL-based optimization

The state set consists of all the combinations by selecting N_1 out of M_1 sensors and N_2 out of M_2 actuators. The size of the state set is calculated by combinations $C(N_1, M_1) \cdot C(N_2, M_2)$. The set of actions for each state can be easily formed by unselected sensors and actuators, which is of size $(N_1+N_2) \cdot (M_1-N_1+M_2-N_2)$. Due to the high number of states and lack of a reward model, the model-free RL technique temporal difference (TD) learning is used. In this way, storing the transition and reward models is avoided. Moreover, it is proven

that the TD update rule can approximate the maximum likelihood of the optimal state value function when performed on finite data and infinite repeated updating.[37]

Algorithm I describes the RL approach to solve the MISDP. Step 1 initializes the learning parameters. At Step 2, the ε-greedy method in (14) is used to determine the action a_t at each time step t. Step 3 then evaluates the next state and collects rewards. The immediate reward r is selected as the negative value of the best achievable H_∞ performance, which is calculated from performing the controller synthesis using Theorem 1. If the selected sensors and actuators render an uncontrollable or unobservable system, then the reward is set to be negative infinite. If the new state is visited and included in the state value table, the reward is directly obtained from the table. Step 4 updates the state value function by the TD update rule.

As shown in (14), the ε-greedy method selects a random action with the probability ε and determines the action to maximal value with the probability $1-\varepsilon$. This approach provides a trade-off between exploration and exploitation, in other words, a balance of enabling fast convergence and avoiding local maximum. On the contrary, the greedy algorithm is likely to render sub-optimal solutions because it takes a local optimal action a_t^{max} at each step.

$$\pi(a_t|s_t)=\begin{cases}1-\varepsilon+\varepsilon/|\mathbf{A}(s_t)|, \text{if } a=\arg\max\limits_{a\in\mathbf{A}(s_t),s'\in\mathbf{S}} v(s')\\ \varepsilon/|\mathbf{A}(s_t)|, \text{otherwise}\end{cases} \tag{14}$$

Algorithm 1: Pseudo-code of RL algorithm for co-design

Result: optimal sensor/actuator set $S^*=\{S_y,S_u\}$
Step 1. Initialization:
Initialize state value $v(s)$ for all $s\in\mathbf{S}$, the optimal performance index $r_{opt}=-inf$;
Initialize maximum action steps t_{max} in each episode and maximum episodes N_e^{max};
Initialize the learning parameters: discount rate γ_d, learning rate α, exploration probability ε, and ε has a decaying rate ε_d .
while $N_e\le N_e^{max}$ **do**
  while $t\le t_{max}$ **do**
    Initialize exploration probability ε, $t=1$, and state s_0 with all sensors and actuators
    Step 2. Take action a_t by ε-greedy algorithm, remove one sensor or actuator ;
    Step 3. Evaluate next state s_{t+1} and collect immediate reward r_{t+1}
    if *controllable and observable* **then**
      $r_{t+1}\leftarrow-\min\gamma$;
    else
      $r_{t+1}\leftarrow-inf$;
    end
    Step 4. update state value function:
    $v(s_t)\leftarrow v(s_t)+\alpha\left[r_{t+1}+\gamma_d v(s_{t+1})-v(s_t)\right]$;
    if $r_{t+1}>r_{opt}$ **then**
      $r_{opt}\leftarrow r_{t+1}$, $S^*\leftarrow s_{t+1}$;
      break;
    else
      $t\leftarrow t+1$;
    end
  end
  $N_e\leftarrow N_e+1$;
  $\varepsilon\leftarrow\varepsilon\cdot\varepsilon_d$;
end

Results and verification

The total numbers of available sensors and actuators are $M_1=13$, $M_2=9$, and two selection cases are considered. The first case is to select $N_1=9$ sensors and $N_2=6$ actuators, respectively. The second case is to select fewer sensors and actuators $N_1=6$, $N_2=4$. The sizes of state sets in two cases are $C(13,9)\cdot C(9,6)=60060$ and $C(13,6)\cdot C(9,4)=216216$. The state sets are too large to use exhaust search. The bound of scheduling parameter is $\theta\in\Theta=\{80\le\theta(t)\le130\}$, $\dot\theta\in\Lambda=\{-1\le\dot\theta(t)\le1\}$. The weighting factor is $W_{\overline{u}}=100\cdot\mathbf{I}_{M_2}$, and the variance matrix of process disturbance and gust disturbance are both scaled to $0.01\cdot\mathbf{I}$.

The RL approach is implemented according to Algorithm I. The learning parameters in Step 1 are initialized: Discount rate $\gamma_d=0.5$, learning rate $\alpha=0.9$, exploration probability $\varepsilon=1$, and decays in each episode at the rate $\varepsilon_d=0.95$. The maximum action steps t_{max} in each episode is $M_1-N_1+M_2-N_2$ and maximum episode until termination is $N_e^{max}=100$.

Because RL involves random exploration actions in the epsilon-greedy algorithm, the RL is repeated 10 times to explicitly show the average capability and robustness. The best, worst, and mean rewards earned in 10 trials are shown in Figures 4 and 5. Converged solutions can be obtained by TD learning around 10–30 episodes. The optimal sensor and actuator selections and optimal closed-loop performances are summarized in Table 2.

One of the heuristic methods, the greedy algorithm,[39] is compared with RL in solving the discrete optimization. The greedy algorithm is well-known to take local optimal steps and has polynomial time efficiency to arrive at convergence. It has been proved that the algorithm achieves a sub-optimal solution within $(1-[1/e])$ of the optimum.[40]

The selected set is initialized as the set consisting of all sensors and actuators. In each episode, one sensor or actuator is removed greedily and the removal operations are repeated until reaching the expected number of sensors and actuators. The controllability and observability of each sensor/actuator combination need to be checked, and the performance is set to be infinity if not observable and/or not controllable. The results are presented in Figures 6 and 7. Table 3 summarizes the selected sensors, actuators, and optimized closed-loop performance.

For the case of $N_1=9$, $N_2=6$, the greedy algorithm renders the same solution as TD learning. The sensors and actuators near the wing tip are selected because the vibrations near the wing root are less severe than those in other regions on the wing span. Both algorithms will lead to optimal solutions if only very few actions and decisions are sought. In this case, the greedy algorithm has less computation complexity and needs less computational time to obtain a solution.

On the other hand, the greedy algorithm renders a worse solution than TD learning, in the case of $N_1=6$,

$N_2 = 4$. More decisions are included in determining the optimal action sequence. If the set function doesn't satisfy the submodular property, then greedy action at each step will surely deviate from the global optimal solution. TD learning, on the contrary, avoids the local optima, by ε-greedy search and TD updating. If enough iterations are conducted, the solution will converge to the global optima.

This result matches with the expected advantage of ε-greedy in RL over the greedy algorithm. The ε-greedy in (14) renders a balance between exploration and exploitation by randomly choosing the exploration/exploitation actions. Exploitation (greedy algorithm) chooses the best action to get the most reward by exploiting the current knowledge of the sensor/actuator placement. However, being greedy may not actually get the most reward; thus, it may lead to sub-optimal behaviors. Exploration allows an agent to improve its current knowledge about the relationship between sensor/actuator placement and closed-loop control performance, which leads to long-term benefits. The balance between exploration and exploitation can be tuned by the probability variable ε.

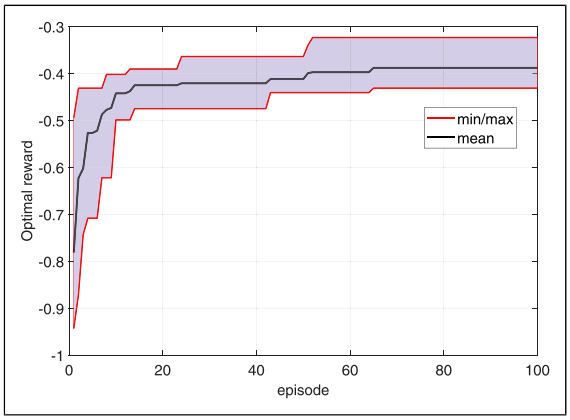

Figure 4. The best/worst and mean reward $-\min \gamma$ versus episode by RL selecting 9 sensors and 6 actuators.

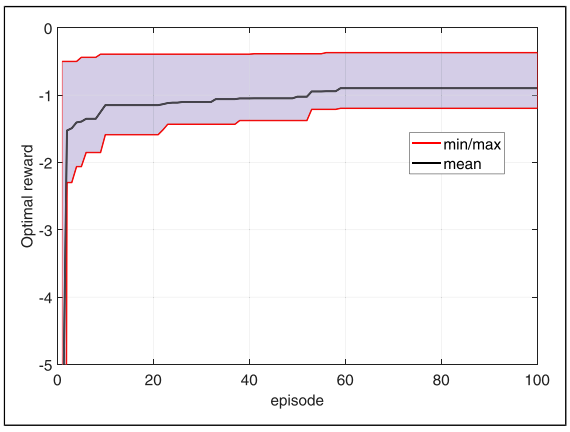

Figure 5. The best/worst and mean reward $-\min \gamma$ versus episode by RL selecting 6 sensors and 4 actuators.

The different solution rendered from the RL and greedy algorithm also indicates that the submodular property doesn't hold for the co-design of OSAP and controllers for the flexible wing. Hence, the greedy algorithm is not recommended if very few sensors and actuators are selected for a nonlinear or parameter-varying system. Besides, the result indicates that installing all sensors and actuators at a few dominant positions is not the best placement policy.

In addition, the GA is used to solve the MISDP and compared with the RL algorithm. The selections of GA parameters are summarized as follows: The probability of cross-over is 0.5; the probability of mutation is 0.1; the population size of each generation is 30; and the elite ratio is 0.1. There are 22 binary variables (13 for sensor placement and 9 for actuator placement). To make a fair comparison, the objective function and evaluation of the co-design are the same as the RL. The results of the best achievable H_∞ performance index γ are shown in Figures 8 and 9. The black curve is the optima in each generation, and the blue curve is the mean value of all populations. It is easy to observe that the GA can obtain the convergent solution in around 30 generations, which needs around 900 evaluations of the co-design variables. The convergent solution of co-design is summarized in Table 4. The convergent solution is the same as that of RL, which can cross-validate with each other.

Verification by high-fidelity simulation

To verify the solutions solved by RL for the co-design problem, the sensor/actuator placement and gain-scheduling control are applied to the high-fidelity simulation of the flexible wing. The high-fidelity model is derived as the full-order model from the finite-element analysis.

Table 2. Summary of the optimal result by the reinforcement learning algorithm.

	$N_1 = 9, N_2 = 6$	$N_1 = 6, N_2 = 4$
Sensor	5, 6, 7, 8, 9, 10, 11, 12, 13	8, 9, 10, 11, 12, 13
Actuator	4, 5, 6, 7, 8, 9	4, 5, 8, 9
γ	**0.3214**	**0.3764**

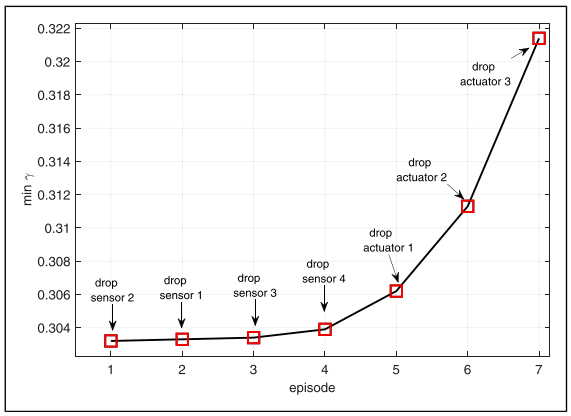

Figure 6. min γ versus episode by the greedy algorithm to select 9 sensors and 6 actuators.

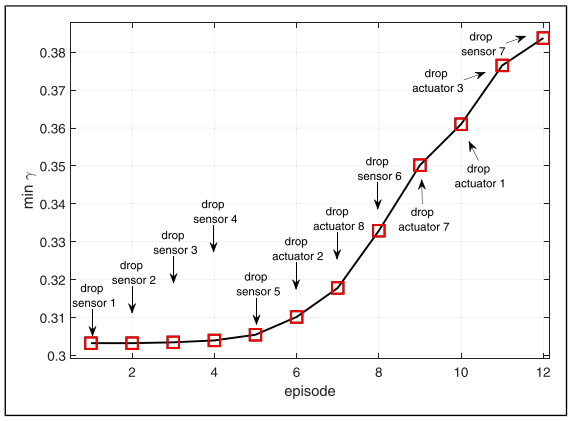

Figure 7. min γ versus episode by the greedy algorithm to select 6 sensors and 4 actuators.

Table 3. Summary of the optimal result by the greedy algorithm.

	$N_1 = 9, N_2 = 6$	$N_1 = 6, N_2 = 4$
Sensor	5, 6, 7, 8, 9, 10, 11, 12, 13	8, 9, 10, 11, 12, 13
Actuator	4, 5, 6, 7, 8, 9	4, 5, 6, 9
γ	**0.3214**	**0.3865**

The high-fidelity simulation scenario is considered as the flexible wing cruising at 10,000 feet with varying flight speeds. The flexible wing will go through a $1 - \cos$ type gust disturbance in Figure 10. The selected sensors and actuators are placed on the full-order model and work together with the gain-scheduling controller to suppress the vibrations caused by gust disturbance. The flight speed envelope is shown in Figure 10. Note that the mode M1 becomes unstable when flight speed is over 115 m/s at time instant $t = 2.2$ second.

Figure 11 plots the vertical displacement at four selected positions on the flexible wing, using the results from RL and the greedy algorithm. First of all, both algorithms produce stabilizing LPV controllers, whereas the open-loop system is unstable. Second, the OSAP and associate

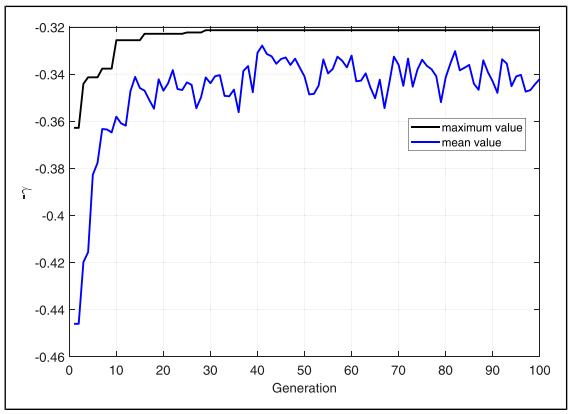

Figure 8. $-\min \gamma$ versus episode by the GA to select 9 sensors and 6 actuators. The blue curve is the mean rewards of all populations in each generation.

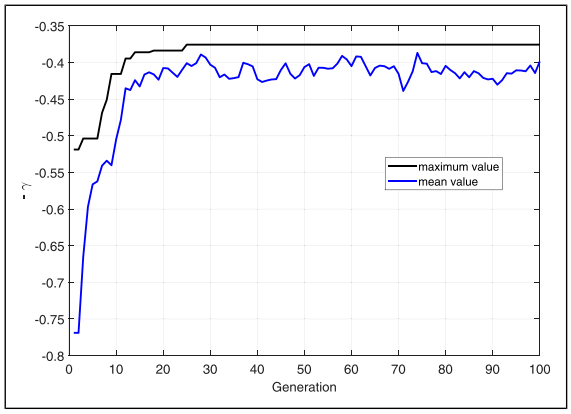

Figure 9. min γ versus episode by the GA to select 6 sensors and 4 actuators. The blue curve is the mean rewards of all populations in each generation.

Table 4. Summary of the optimal result by the genetic algorithm.

	$N_1 = 9, N_2 = 6$	$N_1 = 6, N_2 = 4$
Sensor	5, 6, 7, 8, 9, 10, 11, 12, 13	8, 9, 10, 11, 12, 13
Actuator	4, 5, 6, 7, 8, 9	4,5,8,9
γ	**0.3214**	**0.3764**

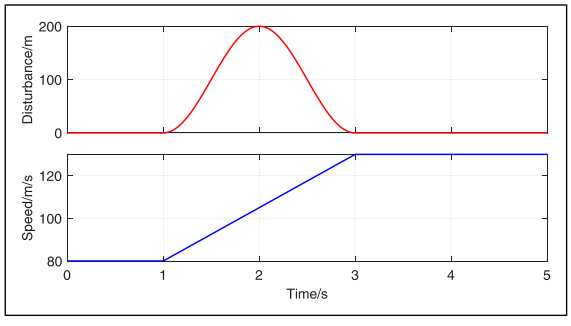

Figure 10. 1-cos type disturbance (top) and flight speed profile (bottom) in simulation scenario.

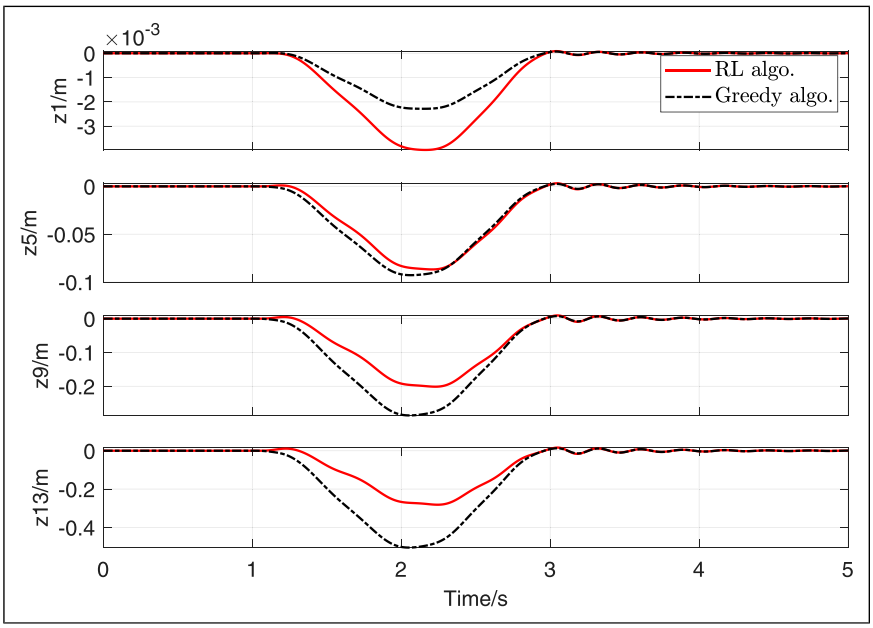

Figure 11. Displacement comparison at wing points, z1 wing root, z5, z9 middle point, and z13 wing tip.

LPV controller from the RL algorithm produces smaller magnitudes at points z5, z9, and z13 (wing tip) than that of the greedy algorithm.

Discussions and conclusion

This paper presents a method using RL to co-design optimal sensor/actuator placement (OSAP) together with H_∞ control for a flexible wing. With a given set of available sensor and actuator locations, the OSAP is formulated to a CO of MISDP, which optimizes the H_∞ performances of the closed-loop system. Reinforcement learning (TD learning) treats sensor/actuator placement as states and solves the CO without relaxing the discrete variables. The effectiveness of RL is demonstrated by the simulation results and closed-loop responses in high-fidelity simulations. The comparisons between the RL, the greedy algorithm, and the GA indicate that the adaptive heuristic RL can lead to better results than the static heuristic greedy method and reduced computations than the GA. The results demonstrate the excellent capability of RL in solving the co-design problem of structure control systems by finding optimal states in high-dimensional search space.

The results also imply the possibility of solving other co-design problems using RL. Future work includes the investigation of computational complexity and the possibility that RL can be implemented safely and in real time. The parameter-varying system, such as morphing wings, provides an application scenario to properly adapt the structure parameters together with an active controller in real time to achieve optimized system performance.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

ORCID iDs

Tianyi He https://orcid.org/0000-0003-3584-9007
Weihua Su https://orcid.org/0000-0002-4458-0524

Note

1. https://github.com/he-tianyi/FlexWingLPVMatrices

References

1. Su W and Cesnik CE. Nonlinear aeroelasticity of a very flexible Blended-Wing-Body aircraft. *J Aircra* 2010; 47(5): 1539–1553. DOI: 10.2514/1.47317
2. Guo C, Lu D, Zhang M, et al. Active control technology for flexible solar array disturbance suppression. *Aeros Sci Tech* 2020; 106: 106148. DOI: 10.1016/j.ast.2020.106148
3. Ouyang Y, Gu Y, Kou X, et al. Active flutter suppression of wing with morphing flap. *Aeros Sci Tech* 2021; 110: 106457. DOI: 10.1016/j.ast.2020.106457
4. Azimi M and Farzaneh Joubaneh E. Dynamic modeling and vibration control of a coupled rigid-flexible high-order structural system: a comparative study. *Aeros Sci Tech* 2020; 102: 105875. DOI: 10.1016/j.ast.2020.105875
5. Gao S and Liu J. Adaptive neural network vibration control of a flexible aircraft wing system with input signal quantization. *Aeros Sci Tech* 2020; 96: 105593. DOI: 10.1016/j.ast.2019.105593
6. Maghami PG and Joshi SM. Sensor/actuator placement for flexible space structures. *IEEE Trans Aeros Electro Sys* 1993; 29(2): 345–351.
7. Skogestad S and Postlethwaite I. *Multivariable feedback control: analysis and design, 2*. New York: Wiley, 2007.

8. Cao Y, Biss D and Perkins J. Assessment of input-output controllability in the presence of control constraints. *Compu chem eng* 1996; 20(4): 337–346.
9. Georges D. *The use of observability and controllability gramians or functions for optimal sensor and actuator location in finite-dimensional systems*. In: Proceedings of 1995 34th IEEE Conference on Decision and Control, Vol 4, New Orleans, 1995. IEEE, pp. 3319–3324.
10. Leleu S, Abou-Kandil H and Bonnassieux Y. Actuators and sensors positioning for active vibration control in flexible systems. In: IFAC Proceedings Volumes 33 (26) (2000) 935 – 940, iFAC Conference on Mechatronic Systems, Darmstadt, Germany, 18-20 September 2000. DOI: 10.1016/S1474-6670(17)39265-0
11. Müller P and Weber H. Analysis and optimization of certain qualities of controllability and observability for linear dynamical systems. *Automatica* 1972; 8(3): 237–246.
12. Tzoumas V, Rahimian MA, Pappas GJ, et al. Minimal actuator placement with bounds on control effort. *IEEE Transa Control Net Sys* 2016; 3(1): 67–78. DOI: 10.1109/TCNS.2015.2444031
13. Gawronski WK. *Dynamics and control of structures: a modal approach*. Springer Science and Business Media, 2004.
14. Liu W, Hou Z and Demetriou MA. A computational scheme for the optimal sensor/actuator placement of flexible structures using spatial h2 measures. *Mech Sys Signal Proc* 2006; 20(4): 881–895, DOI: 10.1016/j.ymssp.2005.08.030
15. Güney M and Eşkinat E. Optimal actuator and sensor placement in flexible structures using closed-loop criteria. *J Sound Vibra* 2008; 312(1–2): 210–233.
16. Hanis T and Hromcik M. Optimal sensors placement and spillover suppression. *Mech sys signal proc* 2012; 28: 367–378.
17. Li D, Li H and Fritzen C. The connection between effective independence and modal kinetic energy methods for sensor placement. *J sound vibra* 2007; 305(4–5): 945–955.
18. Padula SL and Kincaid RK. *Optimization strategies for sensor and actuator placement*, 1999.
19. Singh T, De Mauri M, Decré W, et al. Feedback control of linear systems with optimal sensor and actuator selection. *J Vibra Contr* 2021; 27(11–12): 1250–1264.
20. Joshi S and Boyd S. Sensor selection via convex optimization. *IEEE Transac Signal Proc* 2008; 57(2): 451–462.
21. Jawaid ST and Smith SL. Submodularity and greedy algorithms in sensor scheduling for linear dynamical systems. *Automatica* 2015; 61: 282–288.
22. Bruant I, Gallimard L and Nikoukar S. Optimization of piezoelectric sensors location and number using a genetic algorithm. *Mech Adv Mat Struc* 2011; 18(7): 469–475.
23. Mazyavkina N, Sviridov S, Ivanov Set al. Reinforcement learning for combinatorial optimization: a survey. *Computers and Operations Research* 2021; 134: 105400.
24. Bengio Y, Lodi A and Prouvost A. *Machine learning for combinatorial optimization: a methodological tour d'horizon*. European Journal of Operational Research, 2020.
25. Cappart Q, Goutierre E, Bergman D, et al. Improving optimization bounds using machine learning: Decision diagrams meet deep reinforcement learning. *Proc AAAI Conf Artifi Intelli* 2019; 33: 1443–1451.
26. Kasper K, Mathelin L and Abou-Kandil H. *A machine learning approach for constrained sensor placement*. In: 2015 American Control Conference (ACC), Chicago, 2015, IEEE, pp. 4479–4484.
27. Wang Z, Li H-X and Chen C. Reinforcement learning-based optimal sensor placement for spatiotemporal modeling. *IEEE transac cybernet* 2019; 50(6): 2861–2871.
28. Al-Jiboory AK, Zhu GG, Swei SSM, et al. LPV modeling of a flexible wing aircraft using modal alignment and adaptive gridding methods. *Aeros Sci Tech* 2017; 66: 92–102. DOI: 10.1016/j.ast.2017.03.009
29. He T, Al-Jiboory AK, Zhu GG, et al. Application of ICC LPV control to a blended-wing-body airplane with guaranteed performance. *Aeros Sci Tech* 2018; 81: 88–98. DOI: 10.1016/j.ast.2018.07.046
30. He T, Zhu GG, Swei SS-M, et al. Smooth-switching LPV control for vibration suppression of a flexible airplane wing. *Aeros Sci Tech* 2019; 84: 895–903. DOI: 10.1016/j.ast.2018.11.029
31. Su W and Cesnik CE. Strain-based geometrically nonlinear beam formulation for modeling very flexible aircraft. *Int J Soli Struct* 2011; 48(16–17): 2349–2360.
32. Cesnik CE and Hodges DH. Vabs: a new concept for composite rotor blade cross-sectional modeling. *J Am helicop soc* 1997; 42(1): 27–38.
33. Agulhari CM, Oliveira RCLF and Peres PLD. Robust LMI parser: a computational package to construct LMI conditions for uncertain systems. In: *XIX Brazilian Conference on Automation*. Brazil: CBA 2012Campina GrandePB, 2012, pp. 2298–2305.
34. Apkarian P and Adams RJ. Advanced gain-scheduling techniques for uncertain systems. *IEEE Transac Contr Sys Tech* 1998; 6(1): 21–32. DOI: 10.1109/87.654874
35. Löfberg J. Yalmip: a toolbox for modeling and optimization in MATLAB. In: Proceedings of the CACSD Conference. Taipei, Taiwan, 2004, pp. 284–289.
36. Sturm JF. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimiz Meth Soft* 1999; 11(1): 625–653.
37. Sutton RS and Barto AG. *Reinforcement learning: an introduction*. MIT press, 2018.
38. Xu X, Zuo L and Huang Z. Reinforcement learning algorithms with function approximation: recent advances and applications. *Infor Sci* 2014; 261: 1–31.
39. He T, Zhu GG, Swei SSM, et al. Optimal sensor placement for flexible wings using the greedy algorithm. In: 2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), 2020, pp. 577–582. DOI: 10.1109/AIM43001.2020.9158923
40. Das A and Kempe D. *Submodular meets spectral: greedy algorithms for subset selection, sparse approximation and dictionary selection*, 2011. arXiv preprint arXiv: 1102.3975.

Appendix

The LPV matrices of the flexible wing as a polynomial parameter-dependent matrix are shown in (15). The matrix $A(\theta)$ shows the polynomial parameter dependency, which is plotted in Figure 2. The matrices $B(\theta)$ and $C(\theta)$ are hard

to display here due to high dimension. They can be extracted from the download link. The flight speed is scheduling parameter θ, and it ranges from 80 to 130 m/s

$$A(\theta) = diag(M_1, M_2, M_3, M_4, M_5, M_6),$$
$$M_i = \begin{bmatrix} \alpha_i(\theta) & -\beta_i(\theta) \\ \beta_i(\theta) & \alpha_i(\theta) \end{bmatrix}, \text{for } i = 1, 2, \ldots, 6. \quad (15)$$

$\alpha_1(\theta) = 9.47–0.24\theta+0.0014\theta^2$,
$\beta_1(\theta) = 0.93+0.09\theta+5.6\times10^{-4}\theta^2$;
$\alpha_2(\theta) = -7.91+0.15\theta-0.0012\theta^2$;
$\beta_2(\theta) = 27.47-0.08\theta+0.0012\theta^2$;
$\alpha_3(\theta) = 6.65-0.24\theta+0.0014\theta^2$;
$\beta_3(\theta) = 29.54+0.04\theta+0.0015\theta^2$;
$\alpha_4(\theta) = -5.93-8\times10^{-4}\theta+0.07\times10^{-4}\theta^2$;
$\beta_4(\theta) = 48.52-0.0059\theta+0.53\times10^{-4}\theta^2$;
$\alpha_5(\theta) = 0.69-0.23\theta+0.75\times10^{-4}\theta^2$;
$\beta_5(\theta) = 0.17-0.0047\theta+0.47\times10^{-4}\theta^2$;
$\alpha_6(\theta) = -18.04-0.007\theta+0.22\times10^{-4}\theta^2$;
$\beta_6(\theta) = 82.68 + 0.017\theta-0.50\times10^{-4}\theta^2$.