Aerospace Science and Technology 118 (2021) 107014

Contents lists available at ScienceDirect

Aerospace Science and Technology

www.elsevier.com/locate/aescte

Integration of path planning, trajectory generation and trajectory tracking control for aircraft mission autonomy

Ji Won Woo, Jun-Young An, Moon Gyeang Cho, Chang-Joo Kim *

Department of Aerospace and Information Engineering, Konkuk University, Seoul, Republic of Korea

ARTICLE INFO

Article history:
Received 19 February 2021
Received in revised form 31 May 2021
Accepted 1 August 2021
Available online 9 August 2021
Communicated by Choon Ki Ahn

Keywords:
Mission autonomy
Path planning
Trajectory generator
Trajectory-tracking control
P-RRT*

ABSTRACT

Integrating multiple features for autonomous flight of aircraft requires extensive research over wide categories, high-level of know-how, and experiences. To contribute to these issues, this paper treats the realizable integration techniques of the path planning, trajectory generation, and trajectory-tracking control for the mission autonomy of the aircraft. A collision-free path over the three-dimensional digital terrain is efficiently generated using the P-RRT* algorithm and optimized using the line-of-sight path optimizer proposed in this paper. After additional waypoints are inserted into the planned straight-line path segments, a flyable trajectory for the position and heading angle can be built using the spline trajectory generator with reference to the prescribed velocity distribution. Finally, the incremental backstepping-control design has been adopted to provide accurate trajectory-tracking control. The paper addresses in detail the underlying rationales of the present selection of each technique from the realizability point of view. The integrated designs are validated through simulation studies for a series of autonomous mission scenarios of the rotorcraft. The results show that the P-RRT* algorithm combined with the line-of-sight path optimizer can provide a fast and efficient solution for path planning. Also, the extremely accurate trajectory-tracking performance has been achieved by applying the incremental backstepping control to the flyable trajectory obtained using the spline trajectory generator.

1. Introduction

Mission autonomy has become a demanding technology for manned or unmanned aircraft during the last two decades. The advancement of the related techniques has been directly affected by the remarkable increases in both military and civil applications of the unmanned aerial vehicle (UAV) in the areas of reconnaissance, search and rescue, area mapping, patrolling, geological surveying, and transportation, etc. In this regard, Kendoul [1] thoroughly investigated the required functional structure for the UAV's autonomy and identified three main functional areas consisting of navigation, guidance, and flight control systems (FCS). The navigation intends to provide the aircraft states required by FCS using its functions like situational awareness, perception, state estimation, and sensing, etc. Whereas the guidance system generates the flyable trajectory for FCS through its reasoning and cognizance, mission planning and execution monitoring, path planning, waypoint sequencer, and trajectory generation functions. Finally, FCS realizes the trajectory-tracking function using the information provided by other systems. Among these complex functional areas, this paper mainly focuses on the efficient integrated design of path planning, trajectory generation, and trajectory-tracking control.

There exist few papers handling "path planning", "trajectory generation", and "trajectory tracking control" in an integrated manner to achieve the practical mission autonomy of the aircraft. This point is the main motivation of the present study and the paper mainly focuses on the careful selection of each algorithm required to achieve the main objectives and on the development of efficient techniques required for the integrated design. The related design may require a great effort with high-level know-hows and experiences. There exist many of available path planning algorithms like potential-field method (PFM) [2], probabilistic roadmaps (PRMs) [3], rapidly exploring random tree (RRT) [4], A* algorithm [5], etc. Among them, the RRT algorithm is well known for its fast convergence even in complex and high-dimensional environments. Whereas the optimal path is extremely difficult to generate using RRT. To cope with this major drawback, many of its improved versions like RRT* [6], RRT*-smart [7], P-RRT* [8], and others [9–13] have been developed. RRT* improves the path optimality by applying major features of the tree rewiring and best neighbor search. It has been proved that RRT* can generate the globally asymptotic optimal path through the iterative improvement process. However, RRT* typically presents poor convergence and demands high com-

* Corresponding author.
E-mail address: cjkim@konkuk.ac.kr (C.-J. Kim).

https://doi.org/10.1016/j.ast.2021.107014
1270-9638/

Nomenclature

R	Radial distance
$\phi(*,*)$	Radial basis function
$\mathbf{r}$	Position vector
$\mathbf{r}_c$	Center position vector of radial basis function
R_0	Scale parameter of radial basis function
$h(*,*)$	Terrain height at $(*,*)$ coordinates
U_{rep}	Repulsive potential
U_{att}	Attractive potential
$\mathbf{F}_{rep}$	Repulsive force vector
$\mathbf{F}_{att}$	Attractive force vector
$\mathbf{r}_{target}$	Position vector of target point
$d(*,*)$	Distance function between two points
$\mathbf{r}_o$	Position vector of the nearest obstacle to random sampled node
d_{min}	Distance between $\mathbf{r}_o$ and random sampled node
d^*_{obs}	Circular distance from $\mathbf{r}_o$
$\mathbf{F}_{net}$	Net force vector
$\mathbf{r}_{p-rand}$	Potentially guided random sampled node's position vector
$\mathbf{\Lambda}$	Step-size matrix
$\lambda_x, \lambda_y, \lambda_h$	Step-size factors for each axis
$\mathbf{r}_A$	Current position vector of aircraft
$\mathbf{r}_B$	Position vector of Line-of-sight connectable point
V_{Fs}	Flight speed
N_{wp}	Number of waypoints
t	Time
s	Traveled arc length
τ	Nondimensionalized time variable
$\{a_{j,k}\}_{j=1,k=0}^{j=N_{wp},k=7}$	Spline polynomial coefficients of k-th order and j-th time segment
a	Acceleration
v_c	Rate of climb
γ_C	Flight path angle
$\boldsymbol{\kappa}$	Curvature vector
$\mathbf{u}_t$	Tangent vector
$\mathbf{u}_n$	Normal vector
$\mathbf{u}_b$	Binormal vector
κ	Norm of curvature vector
g	Gravitational acceleration
$\mathbf{u}_g$	Unit vector toward the gravity force
$\mathbf{v}$	Linear velocity vector
u	Component of linear velocity in x-axis body-fixed frame
v	Component of linear velocity in y-axis body-fixed frame
w	Component of linear velocity in z-axis body-fixed frame
$\boldsymbol{\omega}$	Angular velocity vector
p	Component of angular velocity in x-axis body-fixed frame
q	Component of angular velocity in y-axis body-fixed frame
r	Component of angular velocity in z-axis body-fixed frame
$\boldsymbol{\varphi}$	Euler angle vector of aircraft
ϕ	Roll angle
θ	Pitch angle
ψ	Heading angle
$\mathbf{x}$	State vector
$\mathbf{f}$	External force vector
$\mathbf{g}$	Gravitational acceleration vector
$\mathbf{J}$	Moment of Inertia matrix
$\mathbf{m}$	External moment vector
$\mathbf{T}_v$	Linear velocity transformation matrix
$\mathbf{T}_\omega$	Angular rate transformation matrix
$\mathbf{u}$	control input vector
$\boldsymbol{\xi}$	Slack variable vector
$\mathbf{G}$	Control effective matrix
$\mathbf{z}$	Tracking Error vector
$\boldsymbol{\alpha}$	Virtual control vector
V	Control Lyapunov Function
$\mathbf{Q}$	Weight matrix
$\mathbf{\Lambda}_{\boldsymbol{\xi}}$	Weight matrix of slack variable vector
$\mathbf{K}_1, \mathbf{K}_2$	Gain matrix
ζ	Damping ratio
ω	Natural frequency
$\delta_0, \delta_{1C}, \delta_{1S}, \delta_{TR}$	Primary control inputs of rotorcraft with respect to collective, lateral, longitudinal, and tail Rotor

Subscripts

b	Body-fixed reference frame
na	Non affine
0	Current
p	Primary
r	Reference
s	Slack variable
d	Desired

puting time. To overcome these problems, RRT*-smart [7] proposed by Islam et al. RRT*-smart intends to limit the sampling area to accelerate the convergence to the optimal path using the generated initial path, but RRT*-smart still shows poor convergence in finding the initial path. However, many approaches have recently been developed to improve the performance of RRT-based algorithms and overcome their shortcomings [14–19]. The algorithms mentioned above may have various pros and cons in their practical applications for the trajectory-tracking or trajectory-following controls. In this regard, this paper intends to integrate path planning, trajectory generation, and trajectory tracking control in an efficient manner considering the real-time applicability. To achieve this, the P-RRT* algorithm is selected to get a fast-initial path solution and combined with a path optimizer proposed in this paper to obtain a near optimal path.

P-RRT* uses the concept of an artificial-potential field (APF) to intentionally position the randomly sampled nodes toward the target point. Thus, it can provide a much faster initial solution of the path-planning than RRT*. The classical planning algorithm (P-RRT*) is applied to cover applicability to a wide range of planning algorithms. However, the real-time applications of P-RRT* are still extremely difficult due to its time-consuming iterative structure to find a near optimal path. For these reasons, the present paper proposes a line-of-sight path-optimization (LoSPO) technique and combines LoSPO with P-RRT*. In this combined approach, the initial path is rapidly generated using P-RRT* and optimized using LoSPO. Therefore, the near optimal path can be planned in real-time without resorting to time-consuming iterations required by P-RRT*. Efficiency and optimality achievable with the proposed path-planning techniques are validated through the applications over the three-dimensional digital terrain randomly generated with radial basis functions.

Regarding the trajectory-generation, Václav Hlaváč [20] differentiated 'path' and 'trajectory' as follows: "The path consists of the

ordered location of points in the space (either the joint space or the operational space), which the vehicle should follow. Whereas the trajectory is a path plus velocities and accelerations at each point". According to his definition, a trajectory can be obtained by simply allocating the arrival time to each waypoint in a path or designating the speed variation over segment paths. From a trajectory-tracking-control point of view, the generated trajectory should be flyable in the sense that the associated flight parameters like the velocity and acceleration over the trajectory must be well within the operational flight envelope (OFE) of the aircraft. Therefore, it is clear that a flyable trajectory cannot be obtained by simply allocating arrival times to each waypoint due to sharp edges connecting two adjacent segments in the planned path. As an alternative, the present paper adopts the spline-curve-fitting technique to get the required smoothness (or continuity) in the generated trajectory. The spline-trajectory generator in this paper expresses each segment of the trajectory as a polynomial function of the associated non-dimensional time or arc-length variables, with which the differentiation and integration of the trajectory function are straightforwardly formulated. Thus, various flight parameters including load factor, turn radius, rate of climb, and others can be easily computed using the differential geometric theory [21] and used to check the flyability of the generated trajectory.

The selection of an adequate FCS design strategy is another formidable task. For this purpose, various factors like the trajectory-tracking performance, robustness to disturbances or uncertainties, dynamical properties of the systems, and operating range of the aircraft should be considered. In addition, the real-time applicability and associated design workloads are also extremely important issues. In this regard, historical advancements in selectable strategies have been well compared in references [22–24]. This paper adopts the incremental backstepping control (IBSC) design to cope with the strong nonlinear inter-axis coupling among flight-dynamic modes and the wide operating range of the aircraft. The stability of IBSC in the trajectory-tracking control is guaranteed overall OFE by the Lyapunov stability criteria. Recently, Lee et al. proposed an integrated nonlinear-control-design technique for the underactuated nonminimum-phase system using the slack variables approach [25], with which the design workloads can be greatly reduced. This paper adopts this approach to IBSC and derives the resultant error dynamics to propose a straightforward design process of the controller parameters. As a result, control design parameters in IBSC can be determined based on the linear control theory. Whereas, the consistent trajectory-tracking performance over OFE is achievable without resorting to the traditional gain-scheduling strategy. It is well known that the performance of IBSC is not affected by the mismatched uncertainties. In a case when the control effectiveness matrices obtained using the linearized model are works well, IBSC can be implemented without installing the complex and time-consuming nonlinear aircraft model in the flight-control computer. This point is much advantageous for real-time applications with the minimum requirements for the computer resources.

The proposed integration techniques are validated through their applications using the flight dynamics model for the BO-105 helicopter. The selected P-RRT*, spline trajectory generator, and IBSC are some of the advanced techniques and highly recommended for an efficient integrated design. The integrated validation of the effectiveness of these algorithms can be extremely helpful for researchers in this area. The paper is structured as follows. In Section 2, the path planning algorithms are described in detail especially for P-RRT* and LoSPO. Also, the outperformance achievable using the present choices are validated through comparative simulations with other approaches. The spline-trajectory generator is addressed in Section 3 and it uses the waypoint data provided by the path planner. In addition, this section introduces the differentiation and integration formulas for the trajectory function and shows that various flight performance measures to check the flyability can be computed by applying the differential geometric theory. The detailed design of the trajectory-tracking controller based on IBSC is treated in Section 4 with the stability proof. Here, the derivation of error dynamics and the design approach to the controller parameters are addressed. The applications using the Bo-105 math model are presented in Section 5 to validate the overall performance of the proposed techniques. Finally, the important findings of the paper are summarized in the Conclusion.

2. Path planning algorithm

The real world for the aircraft operation is a dynamic environment full of moving and stationary obstacles. This paper limits its applications to the static one simply to focus on the detailed features of the proposed techniques. The terrain obstacle and no-fly-zone (NFZ) are considered for the path planning. For this purpose, the digital terrain model is generated using the radial basis function and NFZs including NOTAM (Notice-to-Airman) are added to mimic the operating environment. After introducing the path planning with P-RRT*, the paper proposes the LoSPO algorithm to optimize the generated initial path and explains how to efficiently integrate LoSPO with P-RRT*.

2.1. Digital terrain and obstacle models

Radial Basis Functions (RBFs) have been widely used for accurate surface curve-fitting because it typically requires a much smaller number of parameters due to its meshless features than other polynomial-based methods [26]. This paper generates the digital terrain model by distributing multiple RBFs in the domain of interest. The terrain surface is computed using the weighted sum of the values of these RBFs, the center of which is designated using the random number generator. The value of RBF depends only on the radial distance $R = \|\mathbf{r} - \mathbf{r}_c\|$ from the center $\mathbf{r}_c = (x_c, y_c)^T$ of each function. Many different types of applicable RBFs are defined in Ref. [26]. Among them, the present paper selects the inverse multi-quadratic RBF because of its realistic terrain-featuring characteristics. Thus, the terrain surface is generated using the following function with N RBFs.

$$h(x, y) = \sum_{j=1}^{j=N} s_j \phi_j(R_j, R_{0j}) \tag{1}$$

Where the sign parameter $s_j (= \pm 1)$ intends to model hills and valleys in the terrain. The inverse multi-quadratic RBF has the following form for $j = 1, 2, \cdots, N$.

$$\phi_j(R_j, R_{0j}) = \frac{1}{\sqrt{R_j^2 + R_{0,j}^2}} \quad \text{with } R_j = \left\|\mathbf{r} - \mathbf{r}_{j,c}\right\|_2 \tag{2}$$

Here, $\mathbf{r}_{j,c}$ and R_{0j}, represent the center position of RBF and scale parameter, respectively.

All parameters used in Eq. (1) and (2) are generated with the random-number generator and normalized to have the following variable range of each parameter and height.

$$\begin{matrix} -1 \le x, x_c \le 1 & 0 \le h \le 1 \\ -1 \le y, y_c \le 1 & 0.002 \le R_0 \le 0.01 \end{matrix} \tag{3}$$

Fig. 1 represents the generated digital-terrain surface using 562 RBFs and proves that the realistic terrain can be generated using the present method. Any terrain with different features can be generated with the same procedure and used for a specific application purpose of this paper.

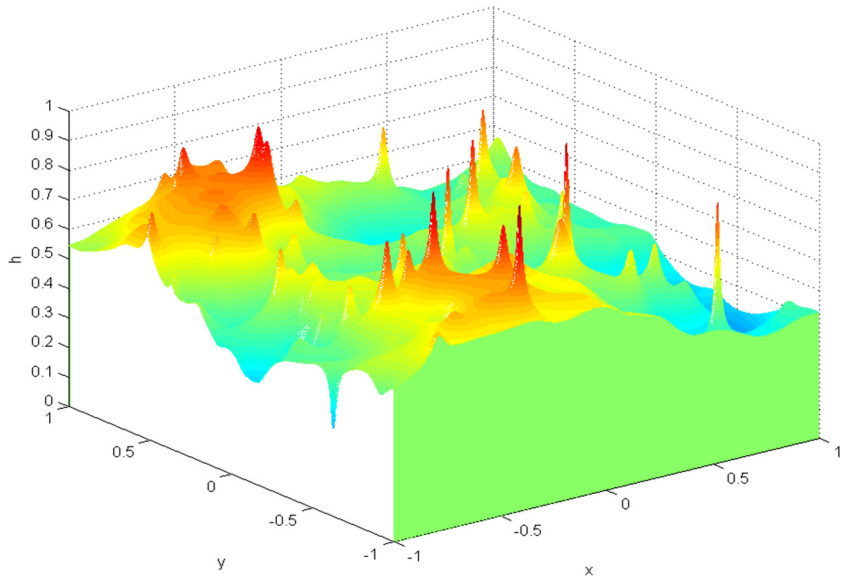


Fig. 1. Terrain surface model generated with radial-basis functions.

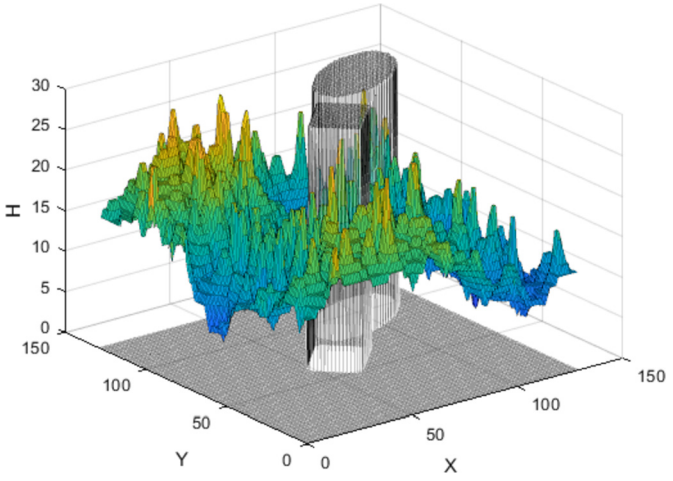


Fig. 2. Generated digital terrain map with NFZ and NOTAM.

Once the digital terrain is generated, various obstacles can be inserted by modifying the height information in the map table to mimic a realistic operating environment of the aircraft. Fig. 2 shows the terrain map in the table size of 129-by-129-by-30, where both NFZ and NOTAM are inserted. The area of NOTAM is represented by a circular column, whereas NFZ is differentiated by the square one. This terrain model is used in the later applications to validate the methods proposed in the paper.

2.2. P-RRT* path-planning algorithm

The Potential function-based RRT* (P-RRT*) algorithm [8] has been used in this paper. P-RRT* generates the force field using the potential function to derive the direction of the path toward the target point without collision with the terrain surface. To achieve this, two artificial potential fields (APF) consisting of the repulsive potential U_{rep} and attractive potential U_{att} are typically used. As depicted in Fig. 3, U_{rep} generated the repulsive force $\mathbf{F}_{rep}$ to avoid the nearest obstacle. Whereas the attractive force $\mathbf{F}_{att}$ generated by U_{att} intends to reposition the sampled node toward the target point. In this way, P-RRT* can achieve much faster path planning than RRT*.

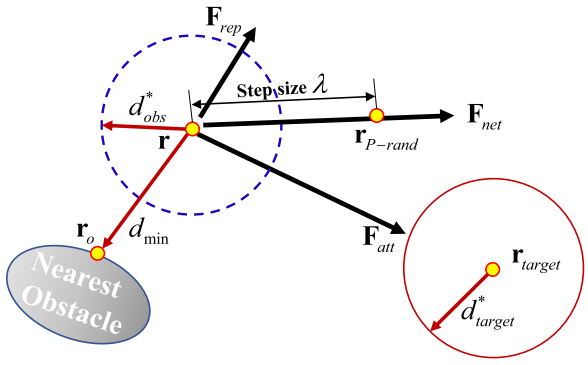


Fig. 3. Schematic view of potential-guided sampling technique.

Qureshi et al. [8] used the following potential shown in Eq. (4) and (5).

$$U_{att} = \begin{cases} K_a d^2(\mathbf{r}, \mathbf{r}_{target}) & (d(\mathbf{r}, \mathbf{r}_{target}) > d_g^*) \\ K_a(d_g^* d(\mathbf{r}, \mathbf{r}_{target}) - (d_{target}^*)^2) & (d(\mathbf{r}, \mathbf{r}_{target}) \le d_g^*) \end{cases} \tag{4}$$

$$U_{rep} = \begin{cases} \frac{1}{2} K_r \left(\frac{1}{d_{\min}} - \frac{1}{d_{obs}^*} \right)^2 & (d_{\min} \le d_{obs}^*) \\ 0 & (d_{\min} > d_{obs}^*) \end{cases} \tag{5}$$

where $\mathbf{r}$, $\mathbf{r}_{target}$, $d(\mathbf{r}, \mathbf{r}_{target})$, and d_{min} are the randomly sampled node, target position, distance function between two points, distance $d(\mathbf{r}, \mathbf{r}_o)$ to the nearest obstacle as defined in Fig. 3, respectively. The parameter K_a and K_r are used to scale the magnitude of each attractive and repulsive potential and d_{obs}^* is typically prescribed as a circular distance to limit the function of the repulsive potential. The forces induced by these potentials can be computed using their gradients as

$$\mathbf{F}_{rep} = -\frac{\partial U_{rep}}{\partial \mathbf{r}}, \quad \mathbf{F}_{att} = -\frac{\partial U_{att}}{\partial \mathbf{r}} \tag{6}$$

The resultant net force $\mathbf{F}_{net}$ is a vector sum of these forces. As a result, the sampled node $\mathbf{r}$ generated with RRT* can be repositioned to a new position $\mathbf{r}_{p-rand}$ using the following formula.

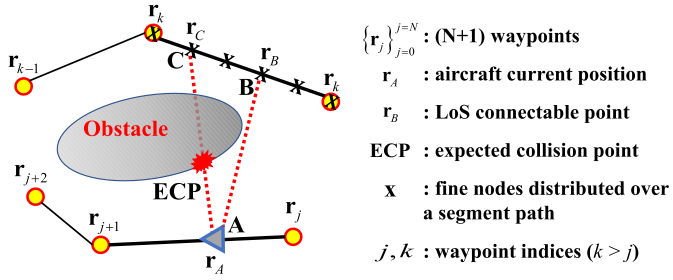
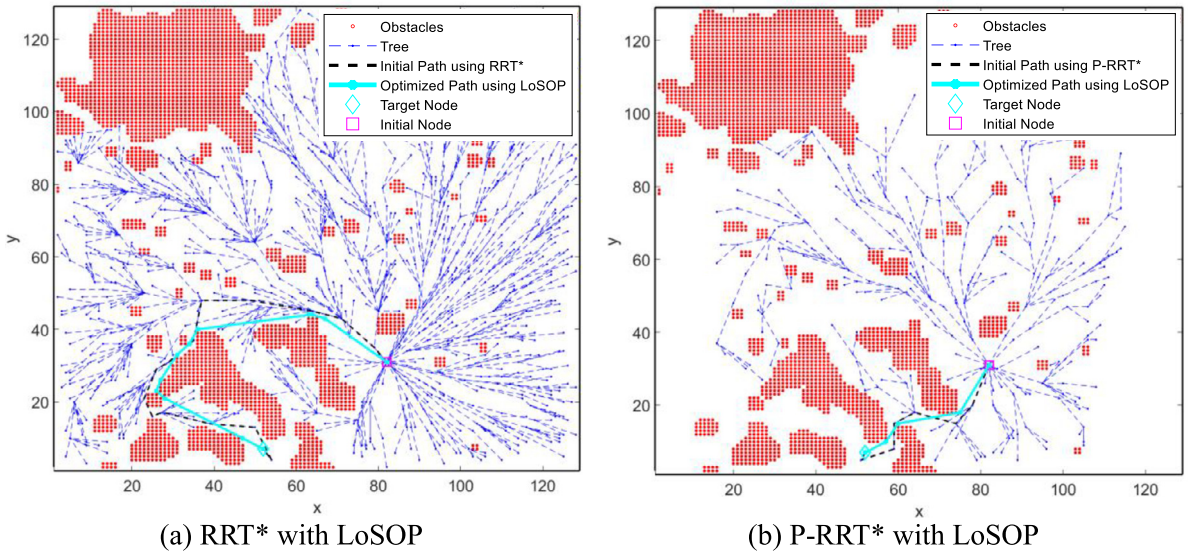


Fig. 4. Schematics of line-of-sight path optimization.



(a) RRT* with LoSOP (b) P-RRT* with LoSOP

Fig. 5. Examples of line-of-sight path optimization.

$$\mathbf{F}_{net} = \mathbf{F}_{att} + \mathbf{F}_{rep}$$
$$\mathbf{r}_{p-rand} = \mathbf{r} + \mathbf{\Lambda}\left(\frac{\mathbf{F}_{net}}{|\mathbf{F}_{net}|}\right) \tag{7}$$
$$\mathbf{\Lambda} = diag(\lambda_x, \lambda_y, \lambda_h)$$

Where the coefficient matrix $\mathbf{\Lambda}$ intends to use different step-size factors for each direction. P-RRT* can obtain a collision-free path by resampling a node if there is an obstacle between $\mathbf{r}_{p-rand}$ and the nearest tree node. In the present application of the P-RRT* algorithm, the step size is set to 15% of each axis size, and the repulsive force is ignored to reduce the computational load. Details on P-RRT* algorithm can be found in [8].

2.3. Line-of-sight path optimization

P-RRT* algorithm can rapidly generate a connecting path up to the target node. As in RRT*, P-RRT* can improve the path optimality by applying the tree-rewiring and best-neighbor-search methods [7,8]. However, such methods typically require time-consuming iterative computations and fail to find a real optimal path even after the extremely large number of iterations because of their trial-and-error based improvement. This paper proposes a blend-new path optimization method named by the line-of-sight path optimization (LoSPO), the schematic view of which is depicted in Fig. 4. LoSPO uses the sequential waypoint data $(\mathbf{r}_0, \mathbf{r}_1, \cdots, \mathbf{r}_N)$ generated through path planning. First, it generates fine nodes (marked **x** in the figure) over a segment path to search for a LOS connectable point $\mathbf{r}_B$, which is defined by a line segment with no expected collision point (ECP-free) from the aircraft position $\mathbf{r}_A$. Once the LOS connectable point $\mathbf{r}_B$ is found, then the waypoint data $(\mathbf{r}_0, \mathbf{r}_1, \cdots, \mathbf{r}_N)$ can be updated by $(\mathbf{r}_0, \cdots, \mathbf{r}_j, \mathbf{r}_A, \mathbf{r}_B, \mathbf{r}_{k+1}, \cdots, \mathbf{r}_N)$. Therefore, the waypoints $(\mathbf{r}_{j+1}, \cdots, \mathbf{r}_k)$ are skipped and the distance to travel can be greatly shortened.

LoSPO achieves the path-optimality by adopting a specific search sequence. With a fixed aircraft position $\mathbf{r}_B$, the search of the LOS connectable point is started from the last segment path defined by two-end points $(\mathbf{r}_{N-1}, \mathbf{r}_N)$. In the same manner, the search over the fine nodes in a segment is performed in reverse order. This process is continued until a LOS connectable point is first found. The optimal path can be constructed by repeating the above processes whenever the aircraft becomes repositioned to the next fine node along its current segment path. The execution of LoSPO is extremely fast even with the relatively large number of waypoints when the number of fine nodes over a segment is reasonably selected considering the desired path resolution. The global optimal path is extremely difficult to obtain and requires high computing time which may hinder its real-time application. The LoSPO proposed in this paper basically uses the planned path with P-RRT*. So, while the initial path optimized with LoSPO is still not a global optimal one, LoSPO provides a near optimal path solution and very effective for real-time application. Therefore, it can be said that the search sequence adopted in LoSPO is crucial to ensure the effective sub-optimal solution with the path segments generated by P-RRT.

The performance of P-RRT* is compared to that of RRT* through their applications to a 2-D map and the results when the connecting path is first found are shown in Fig. 5. The P-RRT* algorithm with the driving potential toward the target node found a much shorter connecting path than RRT* and the required explorations or samples can be greatly reduced with P-RRT* as shown in Fig. 5.

Table 1
Performance of path planning algorithms.

Algorithms	Cost	CPU time (sec)
RRT*	127.2	0.3300
P-RRT*	52.16	0.2820
RRT* with LoSPO	101.2	0.4160
P-RRT* with LoSPO	41.72	0.2940

The shortest paths obtained by applying LoSPO are depicted also in the figure to demonstrate the usefulness of LoSPO. Table 1 compares the cost and computing time with different combinations of algorithms. Here, the path cost is defined as the total arc length traveled through the path. The required computing times for all cases are negligibly small with less than 0.5 seconds and around 20% reduction in the path length is obtained by applying LoSPO to both RRT* and P-RRT*. However, LoSPO additionally requires only 0.1 seconds increase in the computing time. The path length with P-RRT* is less than over 50% of that using RRT* for all cases. The above results prove the realizability and effectiveness of the present choice of P-RRT* combined with LoSPO.

3. Generation of flyable trajectory

The waypoint data $\{\mathbf{r}_i = (x_i, y_i, h_i)^T\}_{i=0}^{i=N}$ generated using a path planner contains only the sequence of position vectors from the start to the target point without any time information. Whereas FCS requires the trajectory parameters consisting of the heading angle (or track angle) as well as the position vector. Accordingly, all these trajectory parameters should be represented as a function of time for trajectory-tracking control. A trajectory can be said flyable when all flight states like the velocity and acceleration (or load factor) associated with the prescribed trajectory are well within OFE of the aircraft. In this regard, the trajectory-tracking control is very similar to the inverse-simulation [27,28] in that both of them use the prescribed trajectory. The former intends to generate the control input providing the desired tracking accuracy. Whereas the latter directly computes the control input, with which perfect tracking is granted. Even with the above difference, the formulation of the inverse simulation can provide the requirements of the flyable trajectory. As shown in the formulation of Ref. [28], the control solution of the inverse simulation becomes obtainable only when the trajectory is prescribed to have continuous derivatives up to the third order. To meet this continuity requirements, Ref. [28] generated the required trajectory using the septic spline polynomial. This paper adopts the same spline curve-fitting technique to generating the flyable trajectory for the trajectory-tracking control.

A path segment connecting two adjacent waypoints is a straight-line element. Therefore, in a case when two connected segments show a large difference in their track angles, the spline-curve can generate an unreasonable detouring trajectory. To avoid such an unfavorable situation, some of the additional waypoints need to be inserted in the segment when the distance between two-end waypoints is greater than a specified minimum distance. As a result, each length of all segments becomes less than the minimum specification. In this way, we can get the modified set of waypoints $\{(x_i, y_i, h_i)\}_{i=0}^{i=N_{wp}}$ ready for its application to the trajectory generation. To simplify the generation process, first, it is assumed that the aircraft heading is aligned to the trajectory. Then, the heading angle at a waypoint can be computed using the vector tangent to the trajectory using the trajectory function for $\{(x_i, y_i)\}_{i=0}^{i=N_{wp}}$. Secondly, the arrival times to each waypoint need a designation in prior. For this purpose, the paper indirectly computes these times by specifying the flight speed V_{Fs} along the trajectory. In a case when the trajectory is represented as a function of the traveled arc length s, Eq. (8) can be used to compute the mentioned arrival times. The traveled arc length is trivially computed from the distributed straight segment paths. For an accelerated maneuvering or curved path, a simple iterative method can be used to predict the accurate arc-length and travel time simultaneously. However, the generation process can be further simplified when a mission is performed at a constant speed.

$$V_{Fs} = \frac{ds}{dt}$$
$$t_{j+1} = t_j + \int_{s_j}^{s_{j+1}} \frac{1}{V_{Fs}} ds \tag{8}$$

For the completeness of the paper, the spline-trajectory generator used in Ref. [27] is briefly introduced here. Considering the additional information available for the arrival time and heading angle, the trajectory functions corresponding to the waypoint data $\{x_i, y_i, h_i, \psi_i, t_i)\}_{i=0}^{i=N_{wp}}$ can be built using the spline interpolation as in Ref. [27]. Since each trajectory for the position and heading parameters can be independently generated, only the x-trajectory generation is considered here. The spline polynomial interpolation over $t \in [t_j, t_{j+1}]$ associated with the j-th segment can be represented using the non-dimensional time variable as

$$x(\tau) = \sum_{k=0}^{k=7} a_{j,k}\tau^k, \quad (j = 1, \cdots, N_{wp} - 1)$$
$$\tau = \frac{t - t_{j-1}}{\Delta t_j} \in [0, 1], \quad \Delta t_j = t_j - t_{j-1} \tag{9}$$

The corresponding differentiation and integration formulas can be derived for their later use.

$$\dot{x}(\tau) = \frac{1}{\Delta t_j} \sum_{k=0}^{k=7} k a_{j,k} \tau^{k-1}$$
$$\int_{t_j}^{t} x(\tau) dt = \Delta t_j \sum_{k=0}^{k=7} \frac{a_{j,k}}{k+1} \tau^{k+1} \tag{10}$$

The $8N_{wp}$ spline coefficients $\{a_{j,k}\}_{j=1,k=0}^{j=N_{wp},k=7}$ should be determined using $\{x_j\}_{j=0}^{j=N_{wp}}$ and the two-end states, which are typically prescribed for the following six conditions.

$$\dot{x}(t_0) = \dot{x}_0,\ \ddot{x}(t_0) = \ddot{x}_0,\ \dddot{x}(t_0) = \dddot{x}_0$$
$$\dot{x}(t_f) = \dot{x}_f,\ \ddot{x}(t_f) = \ddot{x}_f,\ \dddot{x}(t_f) = \dddot{x}_f \quad \text{where } t_f = t_{N_{wp}} \tag{11}$$

Next, the function values and continuity conditions at each waypoint are derived as shown in Eq. (12) and Eq. (13).

$$a_{j,0} = x_j (j = 0, 1, \cdots, N_{wp} - 1), \quad \sum_{k=0}^{k=7} a_{N_{wp},k} = x_{N_{wp}} \tag{12}$$

$$\frac{1}{\Delta t_j} \sum_{k=0}^{k=7} k a_{j,k} = \frac{1}{\Delta t_{j+1}} a_{j+1,1} (j = 1, 2, \cdots, N_{wp} - 1)$$
$$\frac{1}{\Delta t_j^2} \sum_{k=0}^{k=7} k(k-1) a_{j,k} = \frac{2}{\Delta t_{j+1}^2} a_{j+1,2} (j = 1, 2, \cdots, N_{wp} - 1)$$
$$\vdots \tag{13}$$
$$\frac{7!}{\Delta t_j^7} a_{j,7} = \frac{7!}{\Delta t_{j+1}^7} a_{j+1,7} (j = 1, 2, \cdots, N_{wp} - 1)$$

Here, Eq. (13) represents the continuity conditions of higher derivatives from the first to the sixth ones at each intermediate waypoint. Therefore, all of $\{a_{j,k}\}_{j=1,k=0}^{j=N_{wp},k=7}$ can be computed using $8N_{wp}$ linear algebraic equations (LAE) from Eq. (11) to Eq. (13). In this paper, LAE is solved with a block triangular matrix solver combined with the LU decomposition method, and the trajectory generation can be performed with reasonable computing time for a real-time implementation even with waypoints greater than 500. The spline trajectory in Eq. (9) and associated formulas shown in Eq. (10) are extremely useful to estimate the velocity $\dot{\mathbf{r}} = (\dot{x}, \dot{y}, \dot{z})^T$, acceleration $\ddot{\mathbf{r}} = (\ddot{x}, \ddot{y}, \ddot{z})^T$, and the traveled arc length s. These estimations allow a straightforward computation of the flight parameters like flight speed $V_{Fs}(t) = \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2}$, rate of climb v_c, flight path angle γ_C, and the load factor $\mathbf{n}$ in each direction, with which the flyability of the generated trajectory can be checked through the comparison with the maximum allowable flight performance. In a case when the waypoints are represented with the arclength like $\{(x_i, y_i, h_i, \psi_i, s_i)\}_{i=0}^{i=N_{wp}}$, the differential geometric theory [21] can provide further useful flight information using the following definition of the curvature.

$$\boldsymbol{\kappa}(t) = \frac{d^2\mathbf{r}}{ds^2} = \frac{\ddot{\mathbf{r}}}{V_{Fs}^2} \tag{14}$$

The unit vectors in the osculating plane are defined by

$$\mathbf{u}_t = \frac{\dot{\mathbf{r}}(t)}{V_{Fs}},\ \mathbf{u}_n = \frac{\boldsymbol{\kappa}}{\|\boldsymbol{\kappa}\|},\ \mathbf{u}_b = \mathbf{u}_t \times \mathbf{u}_n \tag{15}$$

Where $(\mathbf{u}_t, \mathbf{u}_n, \mathbf{u}_b)$ represent the tangent, normal, and binormal vectors, respectively. Then, the velocity and acceleration vectors are expressed in the Frenet frame as

$$\begin{aligned} \dot{\mathbf{r}}(t) &= V_{Fs}\mathbf{u}_t \\ \ddot{\mathbf{r}}(t) &= \dot{V}_{Fs}\mathbf{u}_t + V_{Fs}\dot{\mathbf{u}}_t = \dot{V}_{Fs}\mathbf{u}_t + \kappa V_{Fs}^2\mathbf{u}_n \end{aligned} \tag{16}$$

Therefore, the corresponding normal load factor becomes as below.

$$\mathbf{n} = \frac{\kappa V_{Fs}^2}{g}\mathbf{u}_n + \mathbf{u}_g \quad \text{with } \kappa = \|\boldsymbol{\kappa}\| \tag{17}$$

Where g and $\mathbf{u}_g$ denote the gravitational acceleration and the unit vector toward the gravity force, respectively. Therefore, the flyability of the generated trajectories can be determined by comparing various parameters computed using Eq. (14) and Eq. (17) with the maximum allowable performance measures prescribed for an aircraft. In a case when some of them exceed the aircraft performance limit, a local re-planning can be applied to modify the trajectory in an efficient manner. This is another advantage achievable with the present spline-trajectory generator.

4. Trajectory-tracking controller design

The design of a trajectory-tracking control for mission autonomy is another formidable task. Historical advancements of selectable technologies have been well documented in Ref. [22–24]. This section mainly focuses on the design methods used in the paper. The trajectory-tracking control is designed using the incremental backstepping control (IBSC) design framework considering its robustness to the model uncertainties and well-structured design process. IBSC is a Lyapunov based nonlinear control and applicable to the affine nonlinear system. As with other nonlinear-control strategies, the IBSC design for the underactuated nonminimum-phase dynamics requires much higher workloads than for the fully actuated system. To avoid the related difficulties, this paper adopts the slack-variable approach to transform the aircraft dynamics into a fully actuated form [29–32].

First, this paper uses the aircraft motion equations derived in Ref. [28] for the inverse-simulation study because of their straightforward applicability in the IBSC design. The Euler equation shown in Eq. (18) can be represented with the corresponding trajectory parameters using kinematic relations in Eq. (19).

$$\begin{aligned} \dot{\mathbf{v}}_b &= \frac{1}{m}\mathbf{f}_b + \mathbf{g}_b - \boldsymbol{\omega} \times \mathbf{v}_b \\ \dot{\boldsymbol{\omega}} &= \mathbf{J}^{-1}(\mathbf{m}_b - \boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega}) \end{aligned}, \quad \mathbf{v}_b = \begin{pmatrix} u \\ v \\ w \end{pmatrix},\ \boldsymbol{\omega} = \begin{pmatrix} p \\ q \\ r \end{pmatrix} \tag{18}$$

$$\begin{aligned} \mathbf{v}_b &= \mathbf{T}_{\mathbf{v}}\dot{\mathbf{r}} \\ \boldsymbol{\omega} &= \mathbf{T}_{\boldsymbol{\omega}}\dot{\boldsymbol{\varphi}} \end{aligned}, \quad \mathbf{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix},\ \boldsymbol{\varphi} = \begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix} \tag{19}$$

Where the subscript b denotes the body-fixed frame. Thus, the second order dynamics for $\mathbf{r}$ and $\boldsymbol{\varphi}$ can be derived for $\mathbf{x} = (x, y, z, \phi, \theta, \psi)^T$ as in Eq. (20), which is represented by the general nonlinear dynamics in Eq. (21).

$$\ddot{\mathbf{r}} = \mathbf{T}_{\mathbf{v}}^{-1}\left(\frac{1}{m}\mathbf{f}_b + \mathbf{g}_b - \boldsymbol{\omega} \times \mathbf{v}_b\right) - \mathbf{T}_{\mathbf{v}}^{-1}\dot{\mathbf{T}}_{\mathbf{v}}\dot{\mathbf{r}} \tag{20}$$

$$\ddot{\boldsymbol{\varphi}} = \mathbf{T}_{\boldsymbol{\omega}}^{-1}\mathbf{J}^{-1}(\mathbf{m}_b - \boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega}) - \mathbf{T}_{\boldsymbol{\omega}}^{-1}\dot{\mathbf{T}}_{\boldsymbol{\omega}}\dot{\boldsymbol{\varphi}}$$

$$\ddot{\mathbf{x}} = \mathbf{f}_{na}(\mathbf{x}, \dot{\mathbf{x}}, \mathbf{u}_p) \tag{21}$$

Here, the subscript in $\mathbf{f}_{na}$ is intentionally used to denote the non-affine force vector in control and $\mathbf{u}_p \in R^4$ represents the primary control of the aircraft.

Incremental dynamics are commonly adopted to mitigate the model dependency in the designed nonlinear controller. Using the current states $(\mathbf{x}_0, \dot{\mathbf{x}}_0)$ and control $\mathbf{u}_{p0}$ at $t = t_0$, the dynamics at the next time step $t = t_0 + \Delta t$ can be approximated using the following definitions.

$$\begin{aligned} \mathbf{x} &= \mathbf{x}_0 + \Delta\mathbf{x} \\ \dot{\mathbf{x}} &= \dot{\mathbf{x}}_0 + \Delta\dot{\mathbf{x}} \\ \mathbf{u}_p &= \mathbf{u}_{p0} + \Delta\mathbf{u}_p \\ \ddot{\mathbf{x}}_0 &= \mathbf{f}_{na0} \\ &= \mathbf{f}_{na}(\mathbf{x}_0, \dot{\mathbf{x}}_0, \mathbf{u}_{p0}) \end{aligned} \qquad \begin{aligned} \mathbf{F}_0 &= \frac{\partial \mathbf{f}_{na0}}{\partial(\mathbf{x}, \dot{\mathbf{x}})} \\ \mathbf{G}_0 &= \frac{\partial \mathbf{f}_{na0}}{\partial \mathbf{u}_{p0}} \end{aligned} \tag{22}$$

The first-order Taylor series expansion of the forcing function approximates Eq. (21) at $t = t_0 + \Delta t$ as

$$\ddot{\mathbf{x}} = \mathbf{f}_{na}(\mathbf{x}, \dot{\mathbf{x}}, \mathbf{u}_p) \approx \mathbf{f}_{na0} + \mathbf{F}_0\begin{pmatrix} \Delta\mathbf{x} \\ \Delta\dot{\mathbf{x}} \end{pmatrix} + \mathbf{G}_0\Delta\mathbf{u}_p \tag{23}$$

Under the assumption of extremely small Δt, the second term in the right-hand-side can be ignored and the following incremental dynamics are obtained.

$$\ddot{\mathbf{x}} = \ddot{\mathbf{x}}_0 + \mathbf{G}_0\Delta\mathbf{u}_p \tag{24}$$

The linear accelerations are typically measured with accelerometers in the navigation system and the backward difference formula is commonly used to estimate the angular accelerations from the measured angular velocity [33]. Here, it is worth mentioning that Eq. (24) is already affine in control. Therefore, design workloads related to the transformation of Eq. (21) into an associated affine form can be removed in the design process.

This paper adopts the slack-variable approach to further reduce the design workloads [29–32]. For this purpose, the slack-variable vector $\mathbf{u}_s \in R^2$ and the corresponding control effectiveness matrix $\mathbf{G}_s \in R^{6\times 2}$ are added to Eq. (25) to get fully actuated system dynamics. The resultant system dynamics have the following form with the system output $\mathbf{y} \in R^6$.

$$\begin{aligned} \ddot{\mathbf{x}} &= \ddot{\mathbf{x}}_0 + \mathbf{G}\Delta\mathbf{u} + \Delta\boldsymbol{\xi} \\ \mathbf{y} &= \mathbf{x} \end{aligned} \tag{25}$$

Where,

$$\mathbf{G} = (\mathbf{G}_0, \mathbf{G}_s) \qquad \Delta\mathbf{u} = \begin{pmatrix} \Delta\mathbf{u}_p \\ \Delta\mathbf{u}_s \end{pmatrix}$$
$$\Delta\boldsymbol{\xi} = -\mathbf{G}_s \Delta\mathbf{u}_s$$

If $\mathbf{G}_s$ is prescribed to get the nonsingular $\mathbf{G}$, the standard nonlinear control design is applicable to the square system of Eq. (25) with the desired trajectory $\mathbf{y}_d = \mathbf{x}_d$.

The IBSC design is started with the following definitions of output errors.

$$\begin{aligned} \mathbf{z}_1 &= \mathbf{x} - \mathbf{x}_d \\ \mathbf{z}_2 &= \dot{\mathbf{x}} - \boldsymbol{\alpha} \end{aligned} \tag{26}$$

Where the virtual control $\boldsymbol{\alpha}$ denotes the reference velocity vector. The controller and update formula for the slack variables can be obtained by using the following control Lyapunov function (CLF) V and by applying the stability condition $\dot{V} \le 0$.

$$\begin{aligned} V &= \frac{1}{2}\mathbf{z}_1^T\mathbf{Q}\mathbf{z}_1 + \frac{1}{2}\mathbf{z}_2^T\mathbf{z}_2 + \frac{1}{2}\Delta\boldsymbol{\xi}^T\boldsymbol{\Lambda}_{\xi}^{-1}\Delta\boldsymbol{\xi} \\ \dot{V} &= \mathbf{z}_1^T\mathbf{Q}\dot{\mathbf{z}}_1 + \mathbf{z}_2^T\dot{\mathbf{z}}_2 + \Delta\boldsymbol{\xi}^T\boldsymbol{\Lambda}_{\xi}^{-1}\Delta\dot{\boldsymbol{\xi}} \end{aligned} \tag{27}$$

Where $\mathbf{Q}$ and $\boldsymbol{\Lambda}_{\xi}$ are positive definite weight matrices. Using Eq. (26) and Eq. (27), we obtain the following result.

$$\begin{aligned} \dot{V} &= \mathbf{z}_1^T\mathbf{Q}(\mathbf{z}_2 + \boldsymbol{\alpha} - \dot{\mathbf{x}}_d) + \mathbf{z}_2^T(\ddot{\mathbf{x}}_0 + \mathbf{G}\Delta\mathbf{u} + \Delta\boldsymbol{\xi} - \dot{\boldsymbol{\alpha}}) \\ &\quad + \Delta\boldsymbol{\xi}^T\boldsymbol{\Lambda}_{\xi}^{-1}\Delta\dot{\boldsymbol{\xi}} \\ &= \mathbf{z}_1^T\mathbf{Q}(\boldsymbol{\alpha} - \dot{\mathbf{x}}_d) + \mathbf{z}_2^T(\mathbf{Q}^T\mathbf{z}_1 + \ddot{\mathbf{x}}_0 + \mathbf{G}\Delta\mathbf{u} - \dot{\boldsymbol{\alpha}}) \\ &\quad + \Delta\boldsymbol{\xi}^T(\boldsymbol{\Lambda}_{\xi}^{-1}\Delta\dot{\boldsymbol{\xi}} + \mathbf{z}_2) \end{aligned} \tag{28}$$

The application of the control laws and update formula in Eq. (29) satisfies the stability condition as proved by Eq. (30) with the positive definite gain matrices, $\mathbf{K}_1$ and $\mathbf{K}_2$.

$$\begin{aligned} \mathbf{K}_1\mathbf{z}_1 &= -\mathbf{Q}(\boldsymbol{\alpha} - \dot{\mathbf{x}}_d) \\ \mathbf{K}_2\mathbf{z}_2 &= -(\mathbf{Q}^T\mathbf{z}_1 + \ddot{\mathbf{x}}_0 + \mathbf{G}\Delta\mathbf{u} - \dot{\boldsymbol{\alpha}}) \\ \Delta\dot{\boldsymbol{\xi}} &= -\boldsymbol{\Lambda}_{\xi}\mathbf{z}_2 \end{aligned} \tag{29}$$

$$\dot{V} = -\mathbf{z}_1^T\mathbf{K}_1\mathbf{z}_1 - \mathbf{z}_2^T\mathbf{K}_2\mathbf{z}_2 \le 0 \tag{30}$$

Using Eq. (31), the feedback control can be generated by Eq. (32).

$$\begin{aligned} \boldsymbol{\alpha} &= -\mathbf{Q}^{-1}\mathbf{K}_1\mathbf{z}_1 + \dot{\mathbf{x}}_d \\ \dot{\boldsymbol{\alpha}} &= -\mathbf{Q}^{-1}\mathbf{K}_1\dot{\mathbf{z}}_1 + \ddot{\mathbf{x}}_d \end{aligned} \tag{31}$$

$$\begin{aligned} \Delta\mathbf{u} &= -\mathbf{G}^{-1}\left\{(\mathbf{K}_2 + \mathbf{Q}^{-1}\mathbf{K}_1)\dot{\mathbf{z}}_1 + (\mathbf{Q}^T + \mathbf{K}_2\mathbf{Q}^{-1}\mathbf{K}_1)\mathbf{z}_1 + \ddot{\mathbf{x}}_0 - \ddot{\mathbf{x}}_d\right\} \\ \mathbf{u} &= \mathbf{u}_0 + \Delta\mathbf{u} \end{aligned} \tag{32}$$

Therefore, the actual feedback control $\mathbf{u}_p$ is determined by $\mathbf{u}$ in Eq. (32).

The stability of the closed-loop system is proved by Eq. (30). However, the positive-definite weight and gain matrices should be carefully selected to guarantee the desired trajectory-tracking performance. To facilitate the design process, this paper investigates the error dynamics and it is assumed that all matrices are diagonal. Thus, the mentioned matrices are definable as

$$\begin{aligned} \mathbf{K}_1 &= diag(k_{1j})_{j=1}^{j=6} \\ \mathbf{K}_2 &= diag(k_{2j})_{j=1}^{j=6} \end{aligned} \quad \text{and} \quad \begin{aligned} \mathbf{Q} &= diag(q_j)_{j=1}^{j=6} \\ \boldsymbol{\Lambda}_{\xi} &= diag(\lambda_{\xi j})_{j=1}^{j=6} \end{aligned} \tag{33}$$

Using Eq. (25) and Eq. (26), the error dynamics are derived as

$$\ddot{\mathbf{z}}_1 = \ddot{\mathbf{x}}_0 + \mathbf{G}\Delta\mathbf{u} + \Delta\boldsymbol{\xi} - \ddot{\mathbf{x}}_d \tag{34}$$

Substituting $\Delta\mathbf{u}$ in Eq. (32) into Eq. (33), we can obtain the following error dynamics.

$$\ddot{\mathbf{z}}_1 + (\mathbf{K}_2 + \mathbf{Q}^{-1}\mathbf{K}_1)\dot{\mathbf{z}}_1 + (\mathbf{Q}^T + \mathbf{K}_2\mathbf{Q}^{-1}\mathbf{K}_1)\mathbf{z}_1 = \Delta\boldsymbol{\xi} \tag{35}$$

Using the definition of Eq. (33), Eq. (35) results in the independent error dynamics as presented in Eq. (36) and Many of parameters in Eq. (33) can be designed by selecting them to meet the desirable response characteristics of Eq. (37), represented by the damping ratio ζ_j and natural frequency ω_j.

$$\ddot{z}_{1,j} + \left(k_{2,j} + \frac{k_{1,j}}{q_j}\right)\dot{z}_{1,j} + \left(q_j + \frac{k_{1,j}k_{2,j}}{q_j}\right)z_{1,j} = \Delta\xi_j, \quad (j = 1, 2, \cdots, 6) \tag{36}$$

$$\ddot{z}_{1,j} + 2\zeta_j\omega_j\dot{z}_{1,j} + \omega_j^2 z_{1,j} = \Delta\xi_j \tag{37}$$

After rigorous investigations for the reduction of design workloads, this paper recommends the following selection

$$k_{1j} = \frac{\zeta_j\omega_j}{q_j}, \quad k_{2j} = \zeta_j\omega_j, \quad q_j = \frac{1}{(1-\zeta_j^2)\omega_j^2} \tag{38}$$

Therefore, most of the design parameters are straightforwardly determined. However, the weight matrix for slack variables still needs to be designed using the trial-and-error based simulation approach.

The remaining topic to close the underlying methodologies of this paper is the specification of the desired attitude trajectories. The attitude dynamics are internal dynamics unobservable from the system output. Most of the traditional applications of BSC tried to define the attitude trajectories for their use in the attitude-tracking control structure [34–37]. However, the attitude controller in the present IBSC mainly intends to stabilize the internal attitude dynamics. Therefore, this paper simply defines the desired attitude trajectories as

$$\begin{aligned} \phi_d &= \phi_{trim}, \quad \dot{\phi}_d = \ddot{\phi}_d = 0 \\ \theta_d &= \theta_{trim}, \quad \dot{\theta}_d = \ddot{\theta}_d = 0 \end{aligned} \tag{39}$$

As a result, the attitude control in this paper is like the stability-augmentation system (SAS) for manned aircraft. Trial simulations with this SAS-type stabilization provide an excellent tracking performance without any noticeable side-effects. Thus, the tracking-performance of IBSC is nearly insensitive to a specific prescription of the desired attitude as used in previous works of literatures [27,28].

5. Applications

The controller structure proposed in this paper for the aircraft mission autonomy is presented in Fig. 6. First, the path planning is performed using P-RRT* combined with LoSPO using the digital-map data, initial, and final nodes provided by the mission planner. Then, the spline trajectory generator produces a flyable reference trajectory of $(x_r, y_r, h_r, \psi_r)^T$. In this phase, the kinematics shown in Eq. (14)–(17) are investigated to check the flyability. In a case when some of the segment paths are found to be unflyable due to the operating limits of the aircraft, the re-planning of those segment paths may be required. The command filter intends to provide the desirable trajectories for the position, attitude, and heading, represented by $(\mathbf{r}_d, \boldsymbol{\varphi}_d)$. With the present use of the SAS-type stabilization of the internal attitude dynamics shown in Eq. (41), other desired trajectories are simply the same as the corresponding reference trajectories $(x_r, y_r, h_r, \psi_r)^T$. Finally, IBSC is applied to get an excellent trajectory-tracking function.

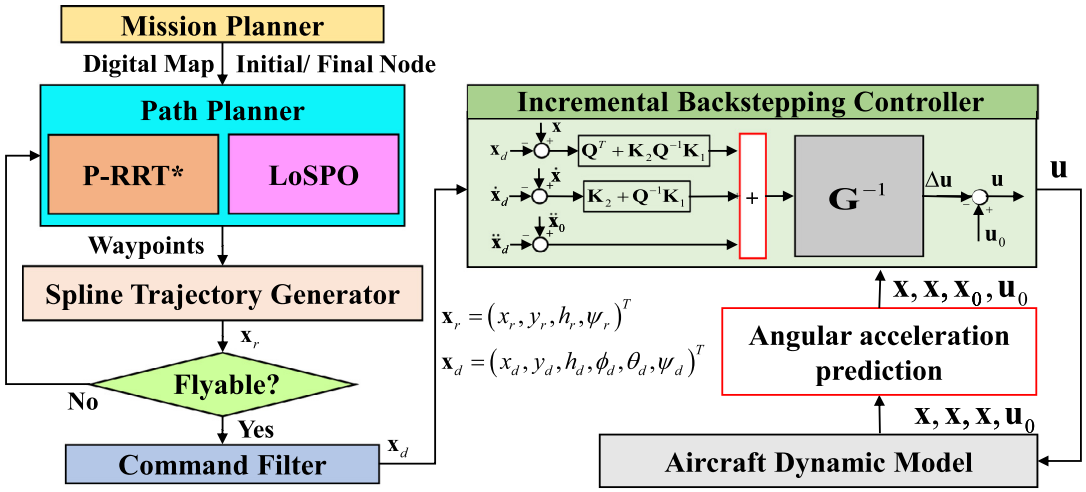


Fig. 6. Trajectory tracking controller structure.

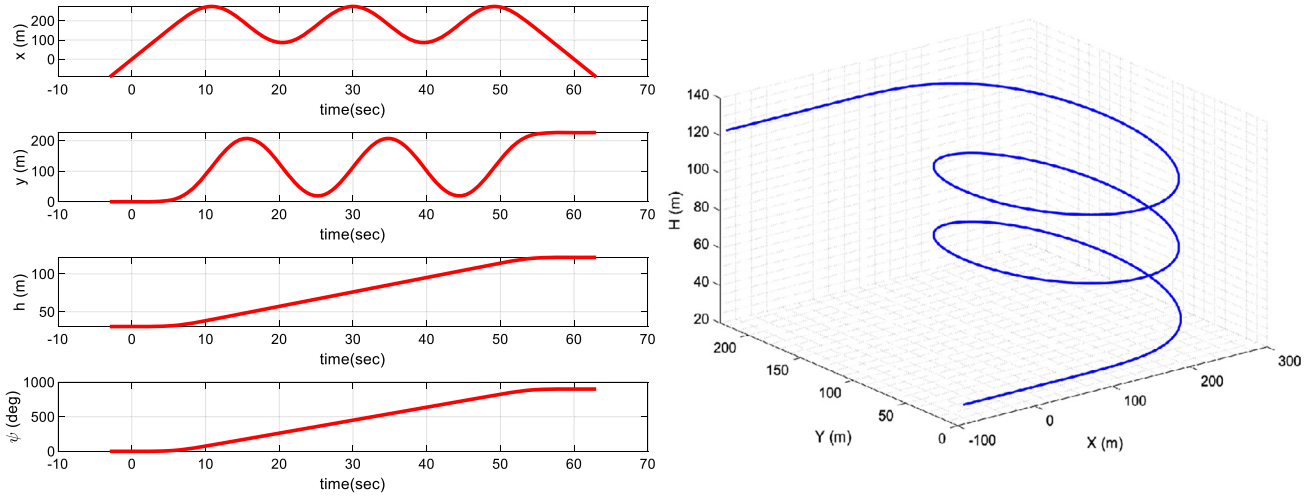


Fig. 7. Generated trajectory for helical-turn maneuver.

The first application is designed to demonstrate the applicability of the present spline-trajectory generator and the trajectory-tracking performance obtainable using IBSC coupled with the SAS-type stabilization of internal dynamics. For this purpose, the helical-turn maneuver is selected for the simulation work. It is an extended version of the transient-turn MTE in ADS-33E-PRF [38] and has an additional 900 degrees turn with a 300 ft increase in altitude. The generated trajectory for 60 knots flight speed is presented in Fig. 7. The details on the related trajectory generation can be found in Ref. [27].

The controller gains are determined using Eq. (38) with the following damping ratios and the natural frequencies for each axis. Therefore, the controller gains and weights are determined as eq. (41) and eq. (42).

$$\zeta_j = 0.75,\ (j = 1, 2, \cdots, 6) \qquad \begin{matrix} \omega_j = 2.0,\ (j = 1, 2, 3, 4) \\ \omega_j = 6.0,\ (j = 5, 6) \end{matrix} \tag{40}$$

$$\mathbf{K}_1 = diag(1.3125, 1.3125, 1.3125, 1.3125, 11.8125, 11.8125)$$
$$\mathbf{K}_2 = diag(1.5, 1.5, 1.5, 1.5, 4.5, 4.5) \tag{41}$$

$$\mathbf{Q} = diag(1.1429, 1.1429, 1.1429, 1.1429, 0.3810, 0.3810) \tag{42}$$
$$\mathbf{\Lambda}_{\xi} = diag(0, 0, 0, 0, 1, 1)$$

It is assumed that accurate linear acceleration and angular rate are available from the navigation sensor system. The required additional angular acceleration for IBSC is predicted simply by applying the backward-difference formula to the angular velocity as in Ref. [33]. These linear and angular accelerations are updated with the 100 Hz sampling rate.

The simulation results are presented in Fig. 8–11. The present design of IBSC with the parameters in Eq. (41)-(42) shows a remarkably excellent trajectory-tracking performance. The trajectory deviations are less than 0.3 m in x- and y-positions. Errors in the altitude and heading trajectories are negligibly small as shown in Fig. 8. The feedback controls generated by IBSC are extremely smooth as shown in Fig. 9 and the roll-attitude angle in Fig. 10 reaches around 40 degrees during the steady climb-turn phase. This point is extremely important in that the present SAS-type stabilization for the internal attitude dynamics works well for the trajectory-tracking control with IBSC under the slack-variable strategy [29–32]. Therefore, the present setting of the desirable attitude trajectories with Eq. (38) may greatly reduce the design workloads because the desirable trajectory-tracking performance is achievable without resorting to the rigorous design for the attitude-command filter as shown in Ref. [39–42].

The next application intends to validate the present integrated design from the path planning to the trajectory-tracking control from the practical realizability point of view. For this purpose, an

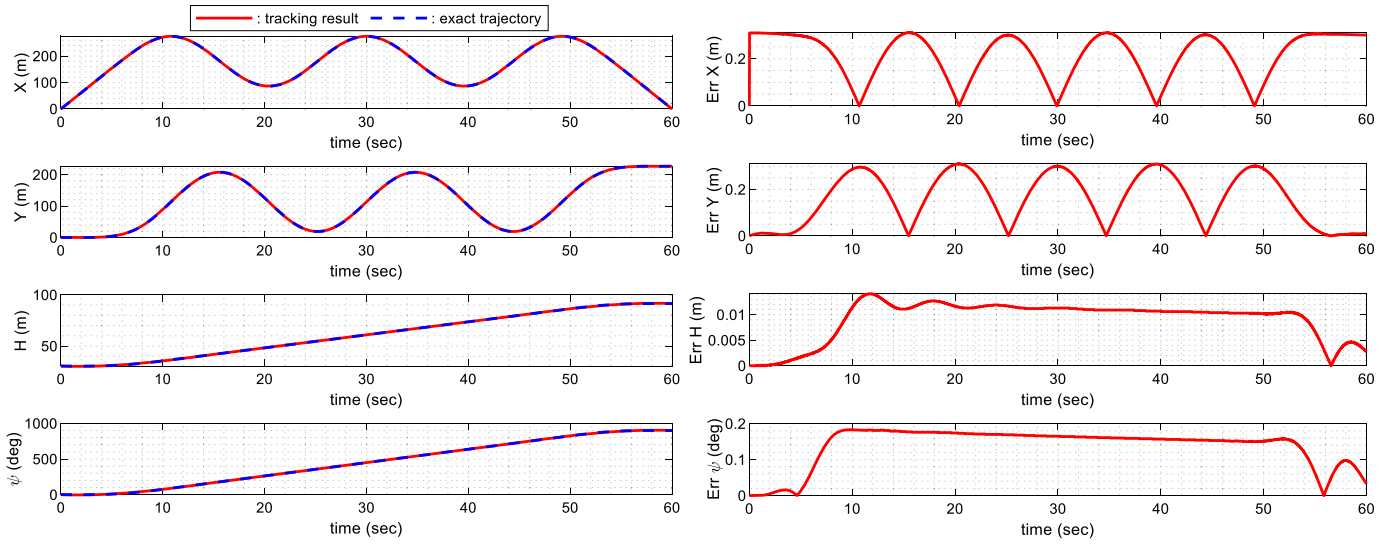


Fig. 8. Simulated trajectories and tracking errors for helical-turn maneuver.

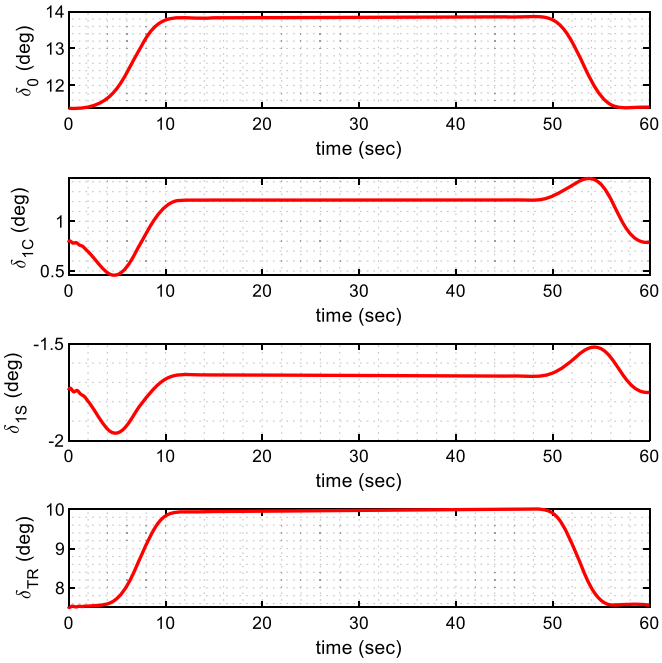


Fig. 9. History of feedback control for helical-turn maneuver.

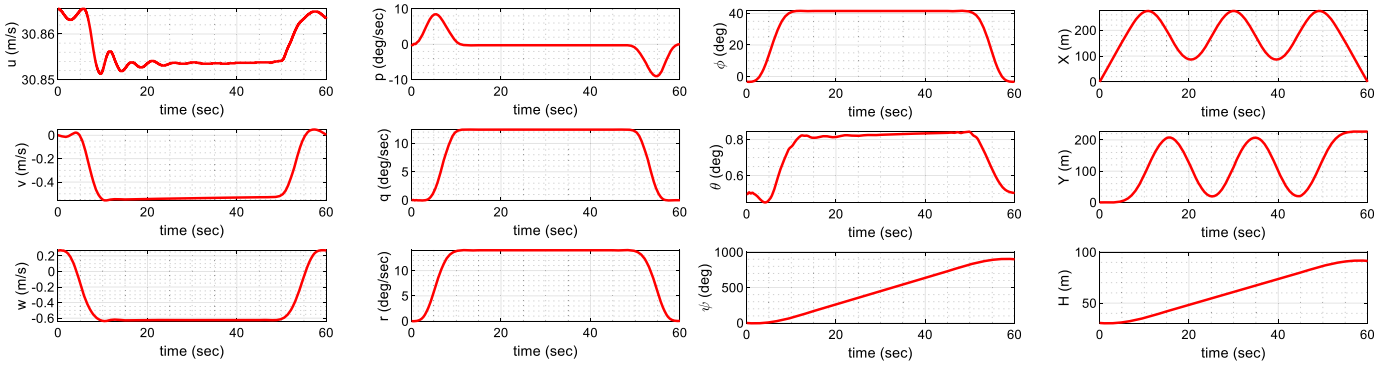


Fig. 10. State variations during the helical-turn maneuver.

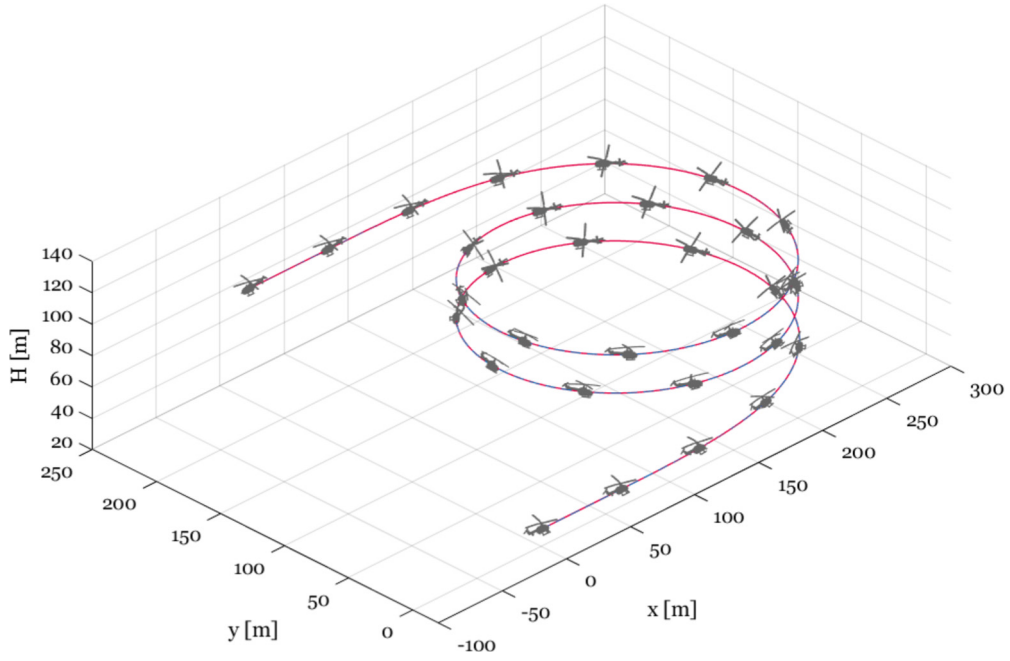
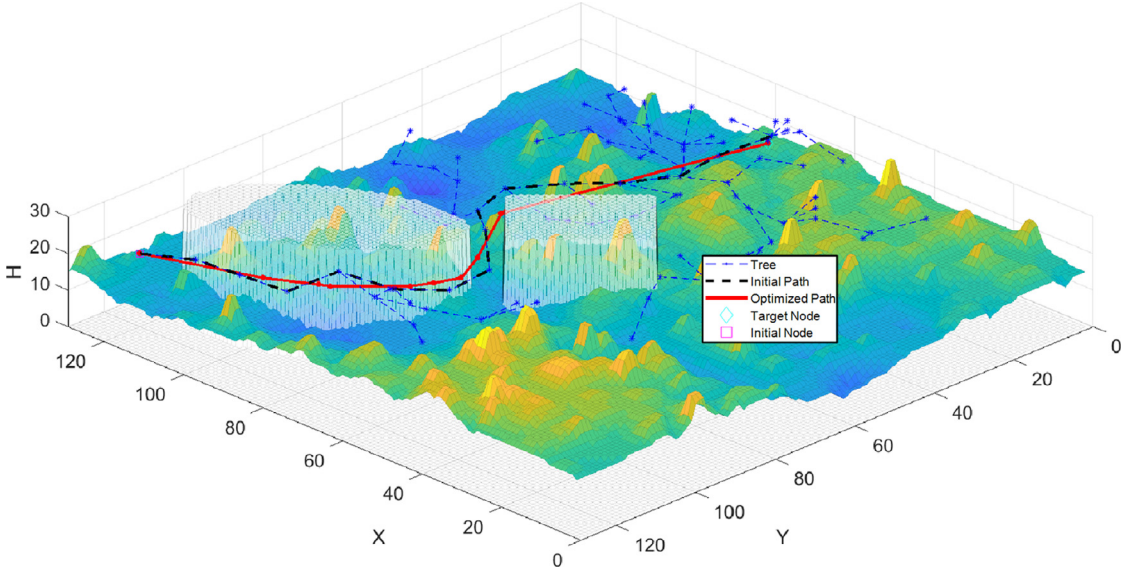


Fig. 11. Visualized flight trajectory for helical-turn maneuver (dashed: aircraft trajectory, solid: generated trajectory).



Initial Node = (120,120,20), Target Node = (74,8,20)
Running time for Initial convergence with P-RRT : 0.037 sec*
Running Time for LoSPO : 0.09100 sec, Total Running Time: 0.1280 sec
Initial Path Cost : 170.7, Optimized Path Cost : 145.2

Fig. 12. Path planning result.

aircraft mission over a 3-D terrain is considered. The planned path is shown in Fig. 12 with the prescribed initial and final nodes while the path planning parameters are described in eq (43). The dash-dotted line describes the path generated using P-RRT*, while the dotted lines depict their search trees. The solid line denotes the optimized final path using LoSPO. The results again prove that the present P-RRT* combined with LoSPO shows a remarkable improvement in the trajectory cost with the negligibly small increase in computational burden. Therefore, it can be claimed that the present path-planning method can allow favorable opportunities for a fast re-planning even in a case when a sudden mission change is demanded by the mission planner. The generated final waypoints with the present algorithm are summarized in Table 2.

$$K_a = 1, \quad d_g^* = 0 \tag{43}$$

The trajectory generated using the spline-trajectory generator is shown in Fig. 13 and Fig. 14, and the time spent in the calculation was around 0.023 seconds. Here, the uniform flight speed of 50 knots is designated over the whole mission flight. The trajectory is smooth enough to provide the required flyability as shown in Fig. 13. A minor correction from the generated path is presented in Fig. 14 for the relatively long straight-line path segment. The main advantage achievable with the spline-trajectory generator is

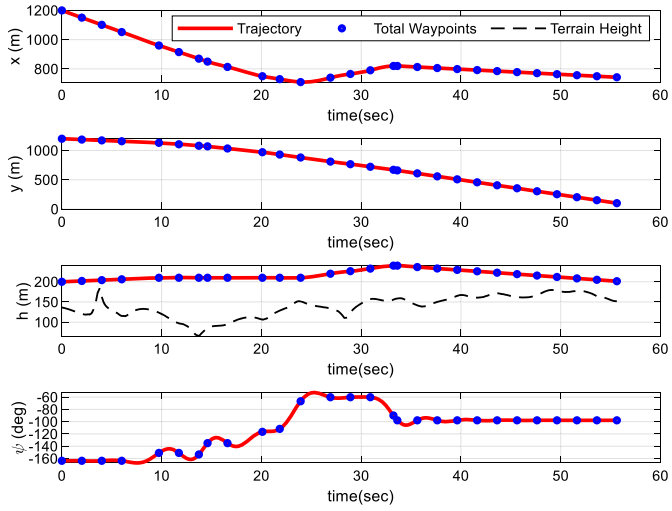


Fig. 13. Generated trajectory.

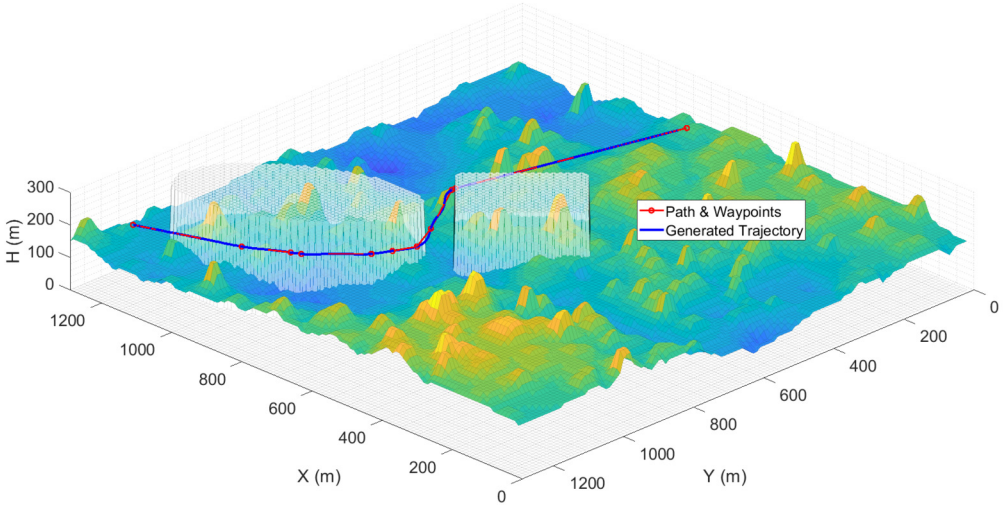


Fig. 14. Generated trajectory on 3D terrain map.

Table 2
Generated waypoint data.

Sequence	X position	Y position	H position
#1	120	120	20
#2	96	113	21
#3	87	108	21
#4	85	107	21
#5	75	97	21
#6	73	93	21
#7	71	88	21
#8	74	81	22
#9	82	67	24
#10	82	66	24
#11	74	8	20

the point that the unfavorable Runge-phenomena in the curve-fitting method like the Lagrange polynomial interpolation can be minimized [43]. The initial and final states including the velocity, acceleration, and jerk are accurately imposed in the present method, which is basically similar to the concept of the Hermite spline interpolation.

The application results of the present IBSC to the generated trajectory are presented in Fig. 15–18. Here, the control effectiveness matrix is generated using two-different strategies. One is directly computed from the nonlinear dynamical HETLAS model during simulation [44,45]. Another one is predicted by adopting the scheduling concept. For this purpose, the linearized models from 0 to 120 knots with a 30-knots speed-interval are computed using HETLAS in prior for various forward speed at the sea level standard condition and used to schedule the control effectiveness matrix during the simulation. As shown in the figures, there exist no discernible differences in the simulated trajectories with two different approaches. From the realizability point of view, this point may indicate that the trajectory-tracking control can preserve its required accuracy without the installation of the complex dynamical model into FCC. Furthermore, a simple scheduling process of the control effectiveness matrix in the linearized models may require much less computing time than the estimation of the control Jacobian from the nonlinear model.

The simulated mission flight contains highly aggressive maneuvers as shown in Fig. 18. The longitudinal and lateral velocities vary over 20 m/sec during the mission flight. The roll attitude temporally reaches over −60 degrees at around 32 seconds and the variation over 40 degrees in the pitch angle is represented at about 25 seconds. The results prove that a consistent trajectory-tracking performance can be maintained using the present IBSC design even with controller gains fixed by Eq. (41) and Eq. (42). Therefore, the design workloads required for the gain-scheduling strategy can be

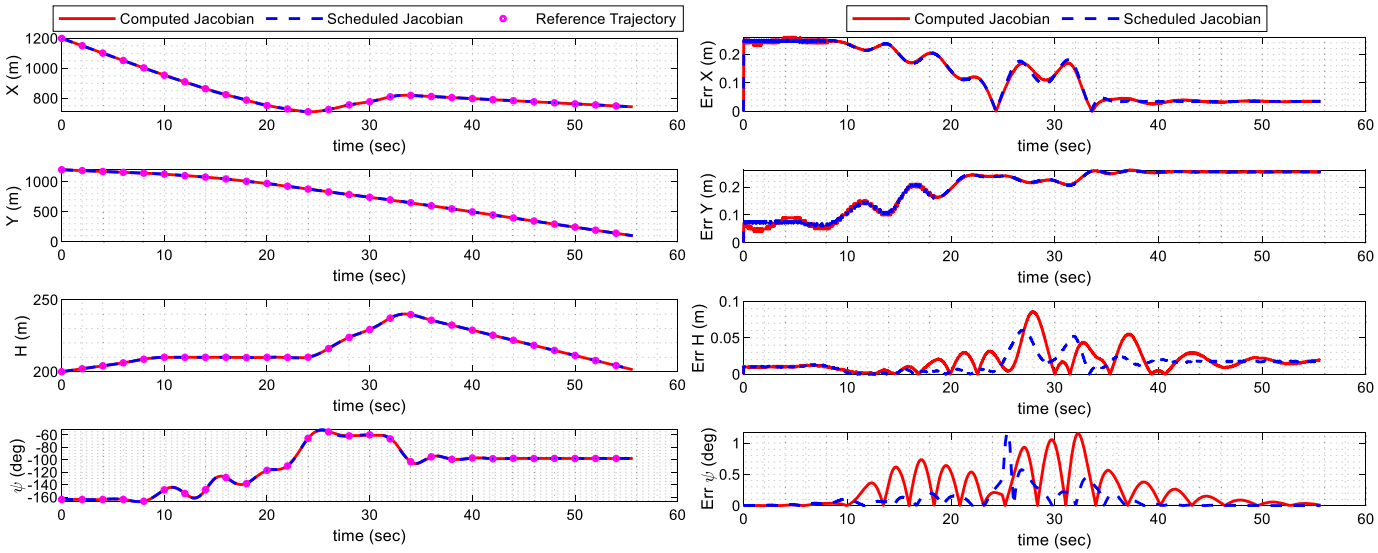


Fig. 15. Trajectory tracking result: tracking performance.

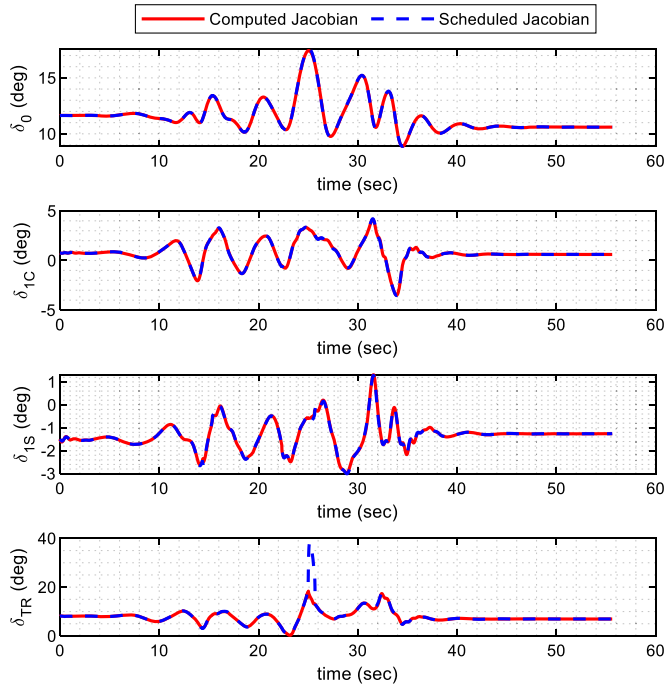


Fig. 16. Trajectory tracking result: control inputs.

further removed by using the integrated slack-variable approach of the present IBSC design.

6. Conclusion

An integrated design approach to path planning, trajectory generation, and trajectory-tracking control has been proposed and validated in this paper for the practical realization of the aircraft mission autonomy. First, the rapidly-exploring random tree algorithm combined with the artificial potential field (P-RRT*) has been adopted to accelerate the initial path-planning process. In addition, the paper proposed the line-of-sight path-optimization (LoSPO) technique to obtain the shortest one with reference to the generated initial path. The effectiveness and real-time applicability of the combined path-planning algorithms were demonstrated through their applications using the two- and three-dimensional digital terrain map. Secondly, the spline trajectory generator has been used to obtain the flyable trajectory, with which the controller allows an excellent trajectory-tracking performance. The paper showed that the main advantage of this method is the straightforward computation of the required velocity and acceleration without the Runge phenomena. Moreover, the spline trajectory generator allows the direct evaluation of the flyability of the generated trajectory using simple kinematical relations. Finally, the incremental backstepping control design was applied for an accurate trajectory-tracking performance without resorting to time-consuming design workloads for the gain scheduling strategy. The integrated controller structure has been addressed in detail to show how each algorithm efficiently works in the integrated systems. The methods proposed in this paper have been validated through a series of simulation investigations.

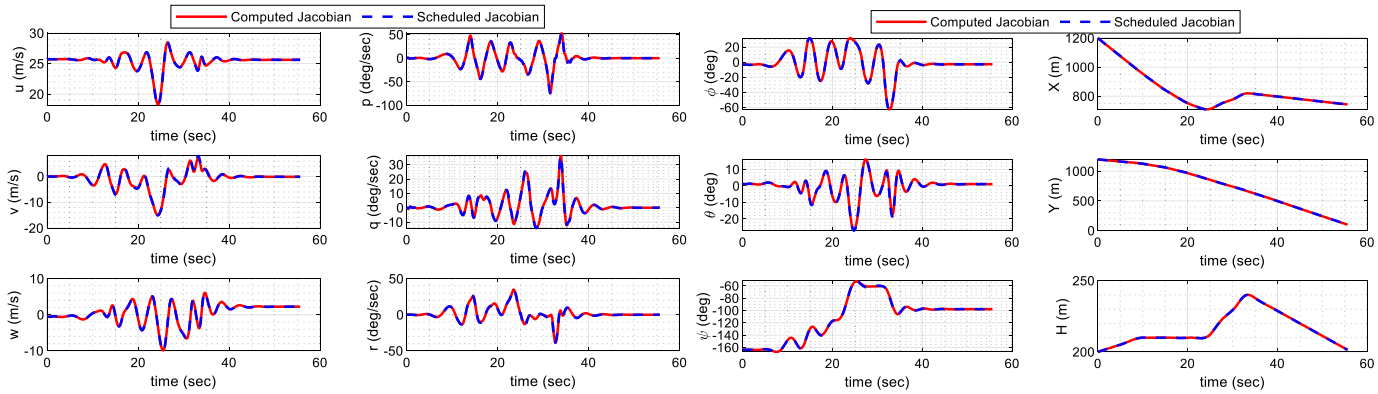


Fig. 17. Trajectory tracking result: states.

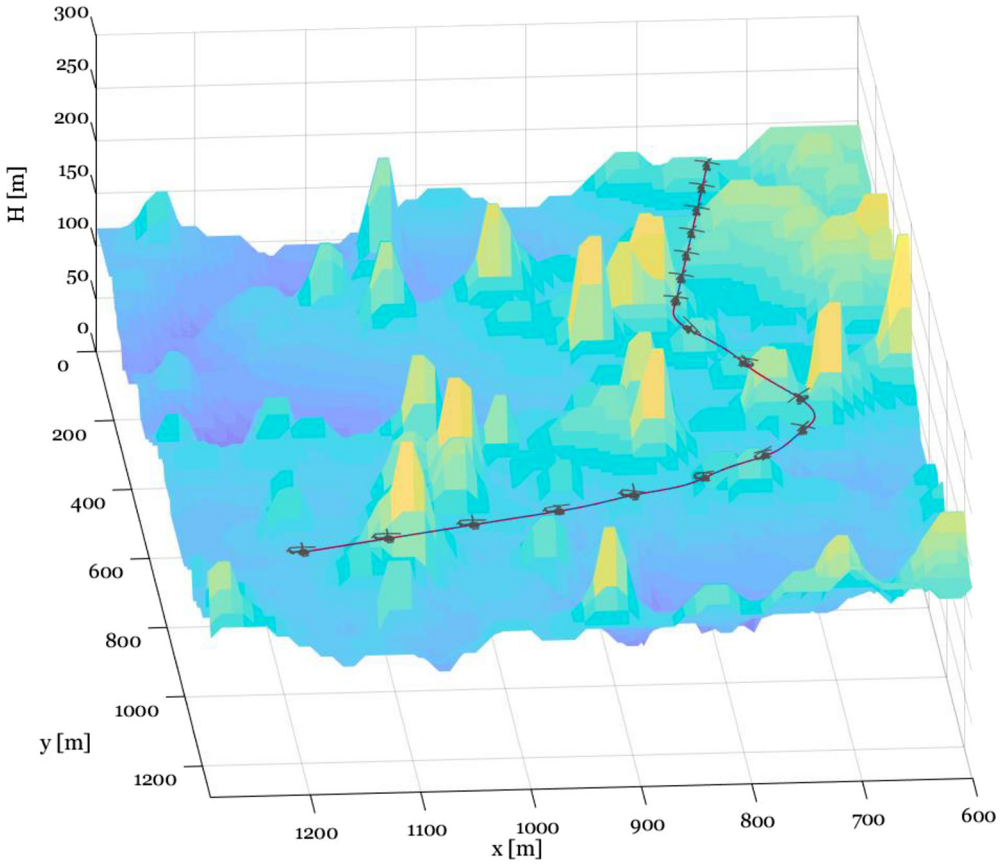


Fig. 18. Trajectory tracking result: visualization.

P-RRT* outperformed RRT* in that it generated over 50% shorter path and required much less computing time than RRT*. P-RRT* combined with LoSPO only spent less than 0.3 seconds for the path planning in the present applications using both complex two- and three-dimensional terrains. Such results prove that the present method can be effectively realizable in the real world and allow favorable opportunities for a fast re-planning even in a case when a sudden mission change is demanded by the mission planner. The present spline-trajectory generator provided smooth flyable trajectories enough to achieve excellent trajectory-tracking accuracy with the present control strategy in various applications. The incremental backstepping control combined with the slack-variable approach showed extremely excellent trajectory-tracking performance. The present stabilization of the internal attitude dynamics is like the conventional stability augmentation system, which can be implemented without any reference attitude trajectories. However, this approach didn't degrade the trajectory-tracking accuracy even for highly aggressive maneuvers with large variations in the pitch and roll attitude angles. In addition, a consistent trajectory-tracking performance could be obtained using the proposed gain selection algorithm for the incremental backstepping control design even with fixed controller gains computed using the proposed formula. Therefore, it can be claimed that the design workloads required for the gain-scheduling strategy can be further removed by using the integrated slack-variable approach of the present paper.

Finally, for the practical realization of the proposed methods, this paper applied two different ways of generating the required control effectiveness matrix. The control Jacobean scheduled using the linearized models showed nearly the same level of the trajectory-tracking accuracy as that computed from the complex

rotorcraft model even for the aggressive maneuvers. From the realizability point of view, this result may indicate that the trajectory-tracking control can preserve its required accuracy without the installation of the complex dynamical model into FCC. A simple scheduling process of the control effectiveness matrix using the linearized models may require much less computing time than the nonlinear counterpart without discernible degradation in the control performance. Therefore, it can be said that the present integrated design approach can be useful for the mission autonomy of future rotary aircraft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. 2020R1A6A1A03046811). Also, this work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2020R1A2C2011955).

References

[1] Farid Kendoul, Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems, J. Field Robot. 29 (1) (2012) 315–378, https://doi.org/10.1002/rob.
[2] Yong K. Hwang, Narendra Ahuja, A potential field approach to path planning, IEEE 8 (1) (1992) 64–75.
[3] L.E. Kavraki, P. Švestka, J.C. Latombe, M.H. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robot. Autom. 12 (4) (1996) 566–580, https://doi.org/10.1109/70.508439.
[4] S.M. LaValle, Rapidly-Exploring Random Trees: a New Tool for Path Planning, 1998.
[5] P.E. Hart, N.J. Nilsson, B. Raphael, Formal basis for the heuristic determination Eijj, Syst. Sci. Cybern. 4 (2) (1968) 100–107.
[6] S. Karaman, E. Frazzoli, Sampling-Based Algorithms for Optimal Motion Planning, 2011.
[7] F. Islam, J. Nasir, U. Malik, Y. Ayaz, O. Hasan, RRT*-Smart: Rapid Convergence Implementation of RRT* towards Optimal Solution, 2012.
[8] A.H. Qureshi, Y. Ayaz, Potential functions based sampling heuristic for optimal path planning, Auton. Robots 40 (6) (2016) 1079–1093, https://doi.org/10.1007/s10514-015-9518-0.
[9] A. Sintov, A. Shapiro, Time-Based RRT Algorithm for Rendezvous Planning of Two Dynamic Systems, 2014.
[10] Z. Tahir, A.H. Qureshi, Y. Ayaz, R. Nawaz, Potentially guided bidirectionalized RRT* for fast optimal path planning in cluttered environments, Robot. Auton. Syst. (2018), https://doi.org/10.1016/j.robot.2018.06.013.
[11] A.H. Qureshi, Y. Ayaz, Intelligent bidirectional rapidly-exploring random trees for optimal motion planning in complex cluttered environments, Robot. Auton. Syst. (2015), https://doi.org/10.1016/j.robot.2015.02.007.
[12] J. Wang, M.Q.H. Meng, O. Khatib, EB-RRT: optimal motion planning for mobile robots, IEEE Trans. Autom. Sci. Eng. (2020), https://doi.org/10.1109/TASE.2020.2987397.
[13] S. Zhang, T. Xu, H. Cheng, F. Liang, Collision Avoidance of Fixed-Wing UAVs in Dynamic Environments Based on Spline-RRT and Velocity Obstacle, 2020.
[14] I.B. Jeong, S.J. Lee, J.H. Kim, Quick-RRT*: triangular inequality-based implementation of RRT* with improved initial solution and convergence rate, Expert Syst. Appl. (2019), https://doi.org/10.1016/j.eswa.2019.01.032.
[15] Y. Li, W. Wei, Y. Gao, D. Wang, Z. Fan, PQ-RRT*: an improved path planning algorithm for mobile robots, Expert Syst. Appl. (2020), https://doi.org/10.1016/j.eswa.2020.113425.
[16] K. Naderi, J. Rajamaki, P. Hamalainen, RT-RRT*: A Real-Time Path Planning Algorithm Based on RRT*, 2015.
[17] K. Yang, S. Moon, S. Yoo, J. Kang, N.L. Doh, H.B. Kim, S. Joo, Spline-based RRT path planner for non-holonomic robots, J. Intell. Robot. Syst., Theory Appl. (2014), https://doi.org/10.1007/s10846-013-9963-y.
[18] H.T.L. Chiang, J. Hsu, M. Fiser, L. Tapia, A. Faust, RL-RRT: kinodynamic motion planning via learning reachability estimators from RL policies, IEEE Robot. Autom. Lett. (2019), https://doi.org/10.1109/LRA.2019.2931199.
[19] J. Wang, W. Chi, C. Li, C. Wang, M.Q.H. Meng, Neural RRT*: learning-based optimal path planning, IEEE Trans. Autom. Sci. Eng. (2020), https://doi.org/10.1109/TASE.2020.2976560.
[20] Václav Hlaváč, Lecture note: robot trajectory generation, http://people.ciirc.cvut.cz/~hlavac/TeachPresEn/55AutonomRobotics/090RobotTrajectoryGenerationEn.pdf, 2020.06.
[21] Barrett O'Neill, Elementary Differential Geometry, 2006.
[22] A. Zulu, S. John, A review of control algorithms for autonomous quadrotors, Open J. Appl. Sci. 04 (14) (2014) 547–556, https://doi.org/10.4236/ojapps.2014.414053.
[23] B. Rubí, R. Pérez, B. Morcego, A survey of path following control strategies for UAVs focused on quadrotors, J. Intell. Robot. Syst., Theory Appl. 98 (2) (2020) 241–265, https://doi.org/10.1007/s10846-019-01085-z.
[24] H. Lee, H.J. Kim, Trajectory tracking control of multirotors from modelling to experiments: a survey, Int. J. Control. Autom. Syst. 15 (1) (2017) 281–292, https://doi.org/10.1007/s12555-015-0289-3.
[25] D. Lee, H.J. Kim, S. Sastry, Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter, Int. J. Control. Autom. Syst. 7 (3) (2009) 419–428, https://doi.org/10.1007/s12555-009-0311-8.
[26] B.S. Morse, T.S. Yoo, P. Rheingans, D.T. Chen, K.R. Subramanian, Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions, in: Proceedings - International Conference on Shape Modeling and Applications, SMI 2001, 2001, pp. 89–98.
[27] C.J. Kim, D.H. Lee, S.W. Hur, Efficient and robust inverse simulation techniques using pseudo-spectral integrator with applications to rotorcraft aggressive maneuver analyses, Int. J. Aeronaut. Space Sci. 20 (3) (2019) 768–780, https://doi.org/10.1007/s42405-019-00160-x.
[28] C.J. Kim, S.H. Lee, S.W. Hur, Kinematically exact inverse-simulation techniques with applications to rotorcraft aggressive-maneuver analyses, Int. J. Aeronaut. Space Sci. 21 (3) (2020) 790–805, https://doi.org/10.1007/s42405-020-00249-8.
[29] R. Padhi, N. Unnikrishnan, S.N. Balakrishnan, Model-following neuro-adaptive control design for non-square, non-affine nonlinear systems, IET Control Theory Appl. (2007), https://doi.org/10.1049/iet-cta:20060364.
[30] D. Lee, H. Lim, H.J. Kim, Y. Kim, K.J. Seong, Adaptive image-based visual servoing for an underactuated quadrotor system, J. Guid. Control Dyn. (2012), https://doi.org/10.2514/1.52169.
[31] M. Kim, Y. Hong, S. Lee, Y. Kim, Slack variables generation via QR decomposition for adaptive nonlinear control of affine underactuated systems, IFAC-PapersOnLine (2016), https://doi.org/10.1016/j.ifacol.2016.09.033.
[32] Y. Hong, Y. Kim, Integrated design of rotary UAV guidance and control systems utilizing sliding mode control technique, Int. J. Aeronaut. Space Sci. (2012), https://doi.org/10.5139/IJASS.2012.13.1.90.
[33] S. Sieberling, Q.P. Chu, J.A. Mulder, Robust flight control using incremental nonlinear dynamic inversion and angular acceleration prediction, J. Guid. Control Dyn. (2010), https://doi.org/10.2514/1.49978.
[34] K. Peng, K.Y. Lum, B.M. Chen, Flight control design with hierarchical dynamic inversion, in: 2010 8th IEEE International Conference on Control and Automation, ICCA 2010, 2010, pp. 297–302.
[35] F. Kendoul, Nonlinear hierarchical flight controller for unmanned rotorcraft: design, stability, and experiments, J. Guid. Control Dyn. (2009), https://doi.org/10.2514/1.43768.
[36] Y. Zou, Nonlinear robust adaptive hierarchical sliding mode control approach for quadrotors, Int. J. Robust Nonlinear Control (2017), https://doi.org/10.1002/rnc.3607.
[37] B. Ahmed, H.R. Pota, Flight Control of a Rotary Wing UAV Using Adaptive Backstepping, 2009.
[38] B.J. Baskett, Aeronautical Design Standard Performance Specification Handling Qualities Requirements for Military Rotorcraft, 2000.
[39] A.N. Kalliny, A.A. El-Badawy, S.M. Elkhamisy, Command-filtered integral backstepping control of longitudinal flapping-wing flight, J. Guid. Control Dyn. (2018), https://doi.org/10.2514/1.G003267.
[40] Y. Liu, J. Ma, H. Tu, Robust command filtered adaptive backstepping control for a quadrotor aircraft, J. Control Sci. Eng. (2018), https://doi.org/10.1155/2018/1854648.
[41] J.O. Birkeland, The concept of autonomy and the changing character of war, Oslo Law Rev. 5 (02) (2018) 73–88, https://doi.org/10.18261/issn.2387-3299-2018-02-02.
[42] Y. He, H. Pei, H. Zhou, Command Filtered Backstepping-Based Flight Control of Unmanned Helicopters with Disturbance Observers, 2015.
[43] C.J. Kim, S. Sung, Efficient ST techniques for nonlinear optimal control analyses using a pseudospectral framework, IEEE Trans. Control Syst. Technol. (2015), https://doi.org/10.1109/TCST.2014.2346994.
[44] Y.H. Yun, C.-J. Kim, K.C. Shin, C.D. Yang, I.J. Cho, Building the Flight Dynamic Analysis Program, HETLAS, for the Development of Helicopter FBW System, 2012.
[45] C.-J. Kim, K.C. Shin, C.D. Yang, I.J. Cho, Interface Features of Flight Dynamic Analysis Program, HETLAS, for the Development of Helicopter FBW System, 2012.