2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

A Flight Envelope Determination and Protection System for Fixed-Wing UAVs

Georgios Zogopoulos-Papaliakos and Kostas J. Kyriakopoulos

Abstract— **In this work we present a novel, approximate, efficient algorithm for determining the Trim Flight Envelope of a fixed-wing UAV, based on a generic, nonlinear numerical model. The resulting Flight Envelope is expressed as a convex intersection of half-spaces. Subsequently, a Model Predictive Controller (MPC) is designed which takes into account the Flight Envelope constraints, to avoid Loss-of-Control. The overall system is shown to operate in real-time in a simulation environment.**

I. Introduction

Unmanned Aerial Vehicles (UAVs) as a technology are past the preliminary stage of being introduced into the society. Even if they still cannot be considered common in everyday life, they already enjoy wide adoption in many commercial sectors, such as agriculture, infrastructure inspection and search and rescue. UAV control is by now a well-studied discipline, but UAV autonomy still remains an open research item.

The immediate goal towards more autonomous UAVs is the design of Fault-Tolerant Controllers (FTC). Faults are a common occurrence in UAVs: They have become large in numbers, most of them are not built to very strict standards and they are subject to a widely unmodelled and dynamic environment. Faults may result in failures, which in turn may be the source of property damage and human injuries, especially as they are introduced into urban environments.

Consequently, there is a need for controllers which can be aware of the system state and status and react appropriately. Apart from maintaining control in the nominal system state, they should adapt to deviations and prevent Loss of Control (LOC) in case of faults. In this work, the scope of research shall be restricted to fixed-wing aircraft platforms.

Another control approach is to make the controller explicitly aware of the performance limits of the aircraft; In the presence of faults, the aircraft model parameters will deviate from their nominal values and the controller should have access to input and state constraints which ensure that the system remains controllable. This set of constraints is commonly termed Flight Envelope (FE).

In traditional aviation literature, the FE is defined as a region of airspeed and altitude where an aircraft is prescribed to operate safely, a closed 2D polytope. That area is usually constrained by limits in the flight altitude, engine performance, overspeed limitations and Angle-of-Attack (AoA) limitations. However, this simplistic definition is suitable only for high-level trajectory planning.

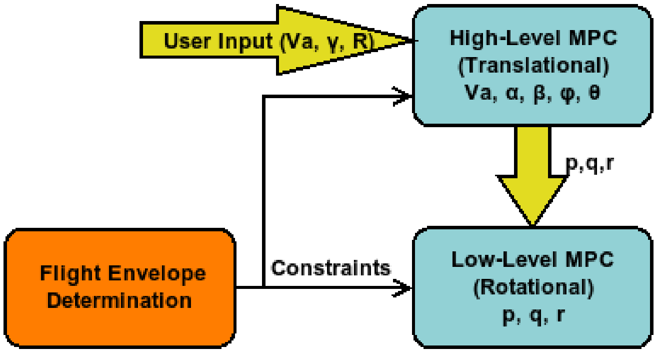


Fig. 1: An overall system diagram

A more useful definition, which can guard against Loss of Control is the set of the state space for which the aircraft can remain controllable.

A first approach towards this set is the trim set or Trim FE, a subspace in the state space where comprized of all of the equilibrium points of the system. For each point in the trim set, a control input exists, subject to input constraints, which ensures that the system will remain in that point.

[1] provided analytical trim equations parameterized for a subset of the state variables for a given simple aircraft model, leading to the trim set. [2] instead opted to use the numerical Newton-Raphson minimization method to calculate the trim states and inputs of a more general aircraft model and proceeded further with characterizing the inherent stability of each equilibrium point by examining eigenvalues of each local, linearized system. In [3], for a linear system including reference commands and disturbances, a polyhedron is calculated offline and projected into a lower subspace in real-time. For given reference command values and estimated disturbances the corresponding achievable Trim FE was given. In [4], a grey-box characterization on a simulated aircraft and control system took place, which was then mapped to a FE spanning the airspeed and the rates of course-over-ground and path-angle.

The aforementioned works provide characterizations which correspond to static trajectories. Instead, a more useful definition is that of Dynamic FE, which corresponds to the subset of the state space where the aircraft is able to operate while maintaining control over its (varying) state. The Dynamic FE is a superset of the Trim FE, since it contains all the transitioning trajectories between trim states. [5] first expressed the reachability problem as an optimal control problem through the evolution of a level set. [6] expressed the level-set problem in a Semi-Lagrangian scheme, taking disturbances into account and representing the FE as a set of Hamilton-Jacobi-Isaacs Partial Differ-

The authors are with Control Systems Lab, School of Mechanical Engineering, National Technical University of Athens, Greece {gzogop, kkyria}@mail.ntua.gr

978-1-7281-7395-5/20/$31.00 ©2020 IEEE

ential equations. The level-set method was also used by [7], where the time-separation principle was considered to isolate only the path-planning variables, including robustness analysis. As late as 2016, [8] stated that real-time numerical flight-envelope estimation was not currently computationally possible through the level-set method, providing a faster approximation approach, at the cost of accuracy. Finally, [9] presents an integrated scheme for LOC prevention through autonomous parameter estimation and analytical FE calculation for a subsystem of the aircraft.

A thorough literature review on the problem of Flight Envelope estimation can be found in [10].

On the other end, design efforts towards controllers that take into account the variable FE have been also made. As early as in [4], the FE was used as constraints in a linearized Model Predictive Control (MPC) problem. In [11], a data-driven identification procedure was used as the basis for safe MPC design. In [12] a linear model-following MPC is presented, incorporating model constraints in the terminal set. In [13], controller design with a region of attraction larger than the FE is presented. More recently, a state transition sequence which remains in the FE is presented in [14].

Many of the FTC approaches employ MPC, because of the straightforward handling of the system model as well as the ability to satisfy input and state constraints. However, the tradeoffs between model fidelity vs time performance and model constraints vs feasibility are an item of active research.

In this work, we present a novel algorithm for the extraction of an approximate yet conservative full-state FE, given a general nonlinear system model. The FE is constructed in the form of a convex intersection of half-spaces, easily incorporated as linear convex constraints in an MPC formulation. A two-layer MPC is designed and its ability to conform to the specified FE is demonstrated through simulation. An abstract diagram of the proposed system can be seen in Figure 1.

II. Flight Envelope Detection

In this work, the Trim Flight Envelope shall be considered. Conceptually, a trim point in the Trim FE is an unaccelerated flight condition such as level flight, constant climb/descent and circular or spiral trajectories. During these maneuvers some of the state derivatives are constant, while others should be zero; these states remain constant.

During unaccelerated flight of a fixed-wing aircraft the derivatives of the lateral position are not of interest. For example during a trimmed turn condition the aircraft performs circular motion. On the other hand, the derivative of the altitude should remain constant, though not necessarily zero.

The other state whose derivative should be constant yet not zero is the yaw angle. The rest of the state derivatives should be equal to zero during a trimmed maneuver.

More formally, the Trim FE can be described as the set of equilibrium points of the aircraft system model:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\zeta}) \tag{1}$$

subject to input constraints $\boldsymbol{u} \in \mathcal{U}$ and with parameters $\boldsymbol{\zeta}$. A point $\boldsymbol{x}^*$ is a trim point if

$$\exists \boldsymbol{u}^* \in \mathcal{U} \quad s.t. \quad \boldsymbol{S} f(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\zeta}) = \boldsymbol{S}\dot{\boldsymbol{x}}^* \tag{2}$$

$\dot{\boldsymbol{x}}^*$ is the vector of system derivatives, with constant and zero elements where appropriate, as described previously. $\boldsymbol{S}$ is a diagonal matrix filtering out the states whose derivatives are not of interest. The parameters $\boldsymbol{\zeta}$ can represent aerodynamic, inertial and propulsion coefficients.

A. Flight Envelope Requirements

In other works where a fast extraction of the Trim FE was achieved, linear models or analytical methods on simplified models were employed. The drawback of these approaches is that they are either inaccurate or rigid against model alterations. In this work we do not settle on a specific aircraft model. Instead, a generic nonlinear model is considered in the form of a numerical computer aircraft simulation library, tailor-made to be modular and flexible.

Additionally, an analytical expression of the FE is required, to be passed afterwards to the MPC. A grid of sampled points is not adequate.

Finally, since contemporary MPC algorithms can run much faster when given convex constraints, we wish the resulting FE expression to be a convex subspace of the state space.

B. Proposed Algorithm

Fast algorithms for contour extraction of 2D and 3D images, such as Active Contours and Morphological Snakes have been proposed([15], [16]). Unfortunately, this class of algorithms are applicable only on 2 and 3 dimensions and some of them operate on an image, i.e. a fully populated sample grid. In this work where we are interested in states spaces of up to 6 variables and an unknown grid, a novel algorithm was required. The proposed algorithm is based on the approximate convex polytope construction algorithm in [17] and is outlined in Algorithm 1. The algorithm requires:

1) An indicator function I which decides if a point is a trim point
2) A set of box constraints $\mathbb{D}$ which loosely enclose the system operational state subspace
3) A vector of desired search resolution for each state space direction $\boldsymbol{e}_f$

The subroutine *Safe Polytope* is defined in Algorithm 2 for clarity. It requires the set of trimmable points, the set of non-trimmable points and a resolution vector.

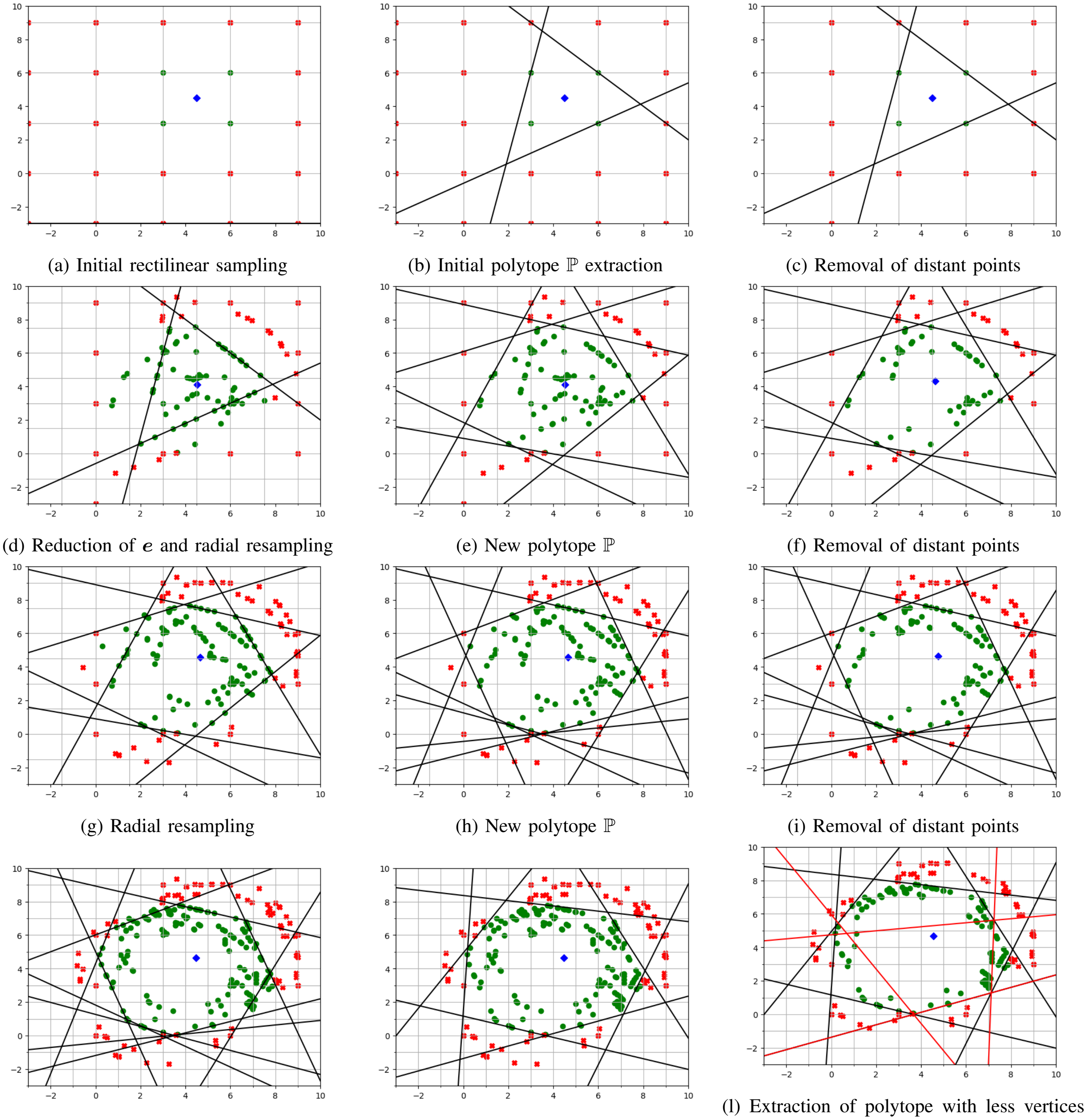
Starting from Algorithm 2, an important assumption for the good operation of the algorithm is that:

Assumption 1: The Trim Flight Envelope, as produced by the indicator function I does not contain holes.

Figure 2 illustrates the Progressive Separation algorithm for a search in 2 dimensions.

The concept behind the proposed algorithm is that it can be tuned to perform faster using some tuning parameters, at the cost of accuracy.

- $\boldsymbol{e}_f$, the final search resolution, serves as the stop condition for Algorithm 1 (Line 9).



(a) Initial rectilinear sampling

(b) Initial polytope $\mathbb{P}$ extraction

(c) Removal of distant points

(d) Reduction of e and radial resampling

(e) New polytope $\mathbb{P}$

(f) Removal of distant points

(g) Radial resampling

(h) New polytope $\mathbb{P}$

(i) Removal of distant points

(j) Reduction of e and radial resampling

(k) New, final polytope $\mathbb{P}$

(l) Extraction of polytope with less vertices (K-Means with 4 clusters)

Fig. 2: Progressive Separation algorithm steps. The indicator function applied is a circle with radius 4 and center at (4,4). Green dots are trim points, red crosses are not.

Algorithm 1 The Progressive Separation Algorithm

```
1:  procedure ProgressiveSeparation($I, \mathbb{D}, \boldsymbol{e}_f$)
2:      Initialize the resolution vector $\boldsymbol{e}$ at some coarse value
3:      Perform sampling on a rectangular grid constrained by $\mathbb{D}$, based on $\boldsymbol{e}$
4:      Use $I$ to initialize the point sets $\mathbf{Y}$ (which satisfy $I$) and $\mathbf{N}$ (which do not)
5:      Find $\boldsymbol{c}$, the centroid of $\boldsymbol{Y}$
6:      V=0
7:      $\mathbb{P}$ = SafePolytope($\boldsymbol{Y}, \boldsymbol{N}, \boldsymbol{e}$)    ▷ Build an initial safe polytope
8:      Set V' as the volume of $\mathbb{P}$
9:      while $\boldsymbol{e} \geq \boldsymbol{e}_f$ do    ▷ While the required resolution hasn't been achieved...
10:         while abs(V'-V)/V $> T$ do    ▷ While polytope volume still changes...
11:             V = V'
12:             Sample $N_s$ points radially from $\boldsymbol{c}$ to the boundary of $\mathbb{P}$ as well as $\boldsymbol{e}$ inside and outside of it.
13:             Update $\boldsymbol{Y}$ and $\boldsymbol{N}$
14:             $\mathbb{P}$ = SafePolytope($\boldsymbol{Y}, \boldsymbol{N}, \boldsymbol{e}$)
15:             V' = Volume($\mathbb{P}$)
16:             Remove any point of $\boldsymbol{Y}$ and $\boldsymbol{N}$ which is further than $\boldsymbol{e}$ from $\mathbb{P}$
17:         $\boldsymbol{e} = \boldsymbol{e}/2$    ▷ Increase resolution
18:     return $\mathbb{P}$
```

Algorithm 2 The Safe Polytope Algorithm

```
1:  procedure SafePolytope($\boldsymbol{Y}, \boldsymbol{N}, \boldsymbol{e}$)
2:      Construct $\mathbb{P}$, the convex separator of Y ( according to [17])
3:      Find the points of $\boldsymbol{N}$ which are inside of $\mathbb{P}$, $\boldsymbol{N}_I$
4:      while $\boldsymbol{N}_I \neq \emptyset$ do
5:          Select a point $\boldsymbol{n}$ of $\boldsymbol{N}_I$
6:          Add to $\mathbb{P}$ a separating hyperplane normal to $\boldsymbol{c}-\boldsymbol{n}$ at $\boldsymbol{n}$
7:          Update $\boldsymbol{N}_I$
8:      return $\mathbb{P}$
```

- $\boldsymbol{e}$, the starting search resolution controls the initial sampling grid density. Since this grid is rectilinear, a small initial resolution can provide a good start for an approximation of $\boldsymbol{y}$ but at higher dimensions the number of samples becomes too large. A value of 1/4 for each domain variable range is usually adequate.
- T, the percentage change of volume in two consecutive polytope approximations serves as a stopping condition for each resolution stage. Afterwards, the resolution is halved (Line 17), but each element of $\boldsymbol{e}$ should only be diminished as far as the corresponding element of $\boldsymbol{e}_f$.
- N_S, the number of radial sample directions, resulting in $3 \times N_S$ new samples on each call of Line 12.

A notable feature of the algorithm is the deletion of points which are more than $\boldsymbol{e}$ away from the polytope boundary. This reduces the total number of points in $\boldsymbol{Y}$ and $\boldsymbol{N}$, lowering the calculation cost of $\mathbb{P}$ and its volume.

The relatively large number of points is also the reason why popular Convex Hull algorithms haven't been used, opting for the faster but non-exact algorithm in [17].

The final result of the Flight Envelope approximation is the polytope $\mathbb{P}$, expressed as a set of linear constraints:

$$\mathbb{P}: \quad \boldsymbol{A}x \leq 0 \tag{3}$$

The number of constraints (rows of $\boldsymbol{A}$) can be further reduced if required by choosing from vertices of $\mathbb{P}$ and build a new polytope of smaller volume. A clustering algorithm (such as K-Means) has been successfully employed for this purpose.

For the present application of Algorithm 1, the indicator function encompasses an optimization over the trim equation 2, a weight matrix $\boldsymbol{W}$ and a threshold by ϵ:

$$\underset{\boldsymbol{u}}{min}\ \boldsymbol{W}(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{\zeta}) - \boldsymbol{S}\dot{\boldsymbol{x}}) \quad < \quad \epsilon \tag{4}$$

Equation 4 is given a point in state space and checks if there is an input able to trim the system, up to an accuracy of ϵ.

III. Controller Design

In this section, the Model-Predictive Controller design is presented, including system and constraints modeling and optimal control objectives definition. The interested reader can refer to [18] and [19] for more details on model derivation.

A. Rotational Dynamics

At the lowest and fastest level of control, the rotational rates of the UAV are regulated. The state and output of this model is $\boldsymbol{\omega} = [p, q, r]^\intercal$ while its input is the control surface deflections $[\delta_a, \delta_e, \delta_r]^\intercal$. The Angle-of-Attack α and Angle-of-Sideslip (AoS) β are external parameters to this model. The aerodynamic coefficients are represented by $C_.$, $\Gamma_.$ and J_y are terms of the matrix of inertia and its inverse, while S, b and c are geometric parameters of the aircraft. $\bar{q} = 0.5V_a^2\rho$ is the dynamic pressure and l, m and n are the aerodynamic moment (torque) components.

$$\begin{aligned}
l = \bar{q}Sb\Big(&C_{l0} + C_{l\beta}\beta + C_{lp}\frac{b}{2V_a}p + C_{lr}\frac{b}{2V_a}r + \\
&C_{l\delta_a}\delta_a + C_{l\delta_r}\delta_r\Big) && \text{(5a)}\\
m = \bar{q}Sc\Big(&C_{m0} + C_{m\alpha}\alpha + C_{mq}\frac{c}{2V_a}q + C_{m\delta_e}\delta_e\Big) && \text{(5b)}\\
n = \bar{q}Sb\Big(&C_{n0} + C_{n\beta}\beta + C_{np}\frac{b}{2V_a}p + C_{nr}\frac{b}{2V_a}r + \\
&C_{n\delta_a}\delta_a + C_{n\delta_r}\delta_r\Big) && \text{(5c)}\\
\dot{p} = &\Gamma_1 pq - \Gamma_2 qr + \Gamma_3 l + \Gamma_4 n && \text{(5d)}\\
\dot{q} = &\Gamma_5 pr - \Gamma_6(p^2 - r^2) + 1/J_y m && \text{(5e)}\\
\dot{r} = &\Gamma_7 pq - \Gamma_1 qr + \Gamma_4 l + \Gamma_8 n && \text{(5f)}
\end{aligned}$$

The constraints of the optimal control problem are normal input constraints

$$u_{min} \leq u \leq u_{max}, \quad u \in \{\delta_a, \delta_e, \delta_t\} \tag{6}$$

as well as the convex, linear flight envelope constraints

$$\boldsymbol{A}[V_a\ \alpha\ \beta\ \phi\ \theta\ r]^\intercal \leq 0 \tag{7}$$

Finally, the cost function for the discretized optimal problem in N steps is defined as

$$J_L = \sum_{1}^{N-1} \boldsymbol{h}_{L,k}^{\intercal} \boldsymbol{Q}_L \boldsymbol{h}_{L,k} + \boldsymbol{h}_{L,N}^{\intercal} \boldsymbol{Q}_{L,N} \boldsymbol{h}_{L,N} \tag{8a}$$

$$\boldsymbol{h}_{L,k} = \left[p_r - p, q_r - q, r_r - r, \delta_a, \delta_e, \delta_r\right] \tag{8b}$$

$$\boldsymbol{h}_{L,N} = \left[p_r - p, q_r - q, r_r - r\right] \tag{8c}$$

where tracked reference quantity is the angular rate $[p_r, q_r, r_r]^{\intercal}$.

B. Translational Dynamics

At the higher level lies the dynamic system involving the translational dynamics. The roll ϕ and pitch θ angles are incorporated in this model, since they contribute to the model of $[V_a, \alpha, \beta]^{\intercal}$ derivatives and depend on the inputs $[p, q, r]^{\intercal}$. In all, the state of the model is $[V_a\ \alpha\ \beta\ \phi\ \theta]^{\intercal}$ and the input vector is $\boldsymbol{\omega}_B = [p\ q\ r]^{\intercal}$.

C_{prop} and k_m are motor parameters and g is the gravity acceleration.

$$F_{lift} = \bar{q}S\left(C_L(\alpha) + C_{Lq}\frac{c}{2V_a}q + C_{L\delta_e}\delta_e\right) \tag{9a}$$

$$F_{drag} = \bar{q}S\left(C_D(\alpha) + C_{Dq}\frac{c}{2V_a}q + C_{D\delta_e}\delta_e\right) \tag{9b}$$

$$\begin{aligned} F_Y = \bar{q}S\Bigg(&C_{Y0} + C_{Y\beta}\beta + C_{Yp}\frac{b}{2V_a}p + \\ &C_{Yr}\frac{b}{2V_a}r + C_{Y\delta_a}\delta_a + C_{Y\delta_r}\delta_r\Bigg) \end{aligned} \tag{9c}$$

$$F_{prop} = 0.5\rho C_{prop}\left((k_m\delta_t)^2 - V_a^2\right) \tag{9d}$$

$$m\dot{V}_a = F_{prop}\cos\alpha\cos\beta - F_{drag} + mg_1 \tag{9e}$$

$$\begin{aligned} m\dot{\beta}V_a = &-F_{prop}\cos\alpha\sin\beta + F_Y + \\ &-mV_a r_W + mg_2 \end{aligned} \tag{9f}$$

$$\begin{aligned} m\dot{\alpha}V_a\cos\beta = &-F_{prop}\sin\alpha - F_{lift} \\ &+mV_a q_W + mg_3 \end{aligned} \tag{9g}$$

$$\dot{\phi} = p + \tan\theta(q\sin\phi + r\cos\phi) \tag{9h}$$

$$\dot{\theta} = q\cos\phi - r\sin\phi \tag{9i}$$

$$\boldsymbol{\omega}_W = \boldsymbol{S}\boldsymbol{\omega}_B \tag{9j}$$

$$[g_1\ g_2\ g_3]^{\intercal} = \boldsymbol{S}\boldsymbol{B}_B\,[0\ 0\ g]^{\intercal} \tag{9k}$$

Note that forces are applied in the Wind-Frame, instead of the Body-Frame. This allows the decoupling of the airdata triplet (V_a, α, β) derivatives from each other and provides better stability and sensitivity for numerical solvers. The rotation from the Body- to the Wind-Frame is performed with the matrix

$$\boldsymbol{S} = \begin{bmatrix} \cos\alpha\cos\beta & \sin\beta & \sin\alpha\cos\beta \\ -\cos\alpha\sin\beta & \cos\beta & -\sin\alpha\sin\beta \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \tag{10}$$

while $\boldsymbol{B}_B$ is the rotation matrix from Earth- to Body-Frame.

The output of the model is $\boldsymbol{y}_H = [V_a\ \gamma\ \dot{\psi}]$, with

$$\dot{\psi} = (q\sin\phi + r\cos\phi)/\cos\theta \tag{11}$$

$$\begin{aligned} \gamma = &\cos\alpha\cos\beta\sin\theta + \\ &-(\sin\phi\sin\beta + \cos\phi\sin\alpha\cos\beta)\cos\theta \end{aligned} \tag{12}$$

The constraints of the optimal control problem are normal input constraints

$$u_{min} \leq u \leq u_{max}, \quad u \in \{p, q, r, \delta_t\} \tag{13}$$

as well as the convex, linear flight envelope constraints

$$\boldsymbol{A}[V_a\ \alpha\ \beta\ \phi\ \theta\ r]^{\intercal} \leq 0 \tag{14}$$

The cost function for the discretized optimal problem in N steps is defined as

$$J_H = \sum_{1}^{N-1} \boldsymbol{h}_{H,k}^{\intercal} \boldsymbol{Q}_H \boldsymbol{h}_{H,k} + \boldsymbol{h}_{H,N}^{\intercal} \boldsymbol{Q}_{H,N} \boldsymbol{h}_{H,N} \tag{15}$$

$$\begin{aligned} \boldsymbol{h}_{H,k} = \Big[&V_{a,d} - V_a, \gamma_d - \gamma, \dot{\psi}_d - \dot{\psi}, \alpha, \beta, p, q, r, \\ &0.5 - \delta_t\Big] \end{aligned} \tag{16}$$

$$\boldsymbol{h}_{H,N} = \left[V_{a,d} - V_a, \gamma_d - \gamma, \phi_r - \phi\right] \tag{17}$$

C. Reference Calculation

Given a desired trajectory vector of airspeed, flight path angle (γ) and turn radius $[V_{a,r}\ \gamma_r\ R_r]^{\intercal}$, the reference for the high-level controller needs to be calculated according to:

$$\dot{\psi}_r = (V_{a,r}/R_r)\cos\gamma_r \tag{18}$$

$$\phi_r = \tan^{-1}(\dot{\psi}_r V_{a,r}/g) \tag{19}$$

The trajectories of the intermediate inputs $[\delta_t\ p\ q\ r]^{\intercal}$ are passed to the low-level controller for tracking.

Between the rotational and translational model there are in total 8 state variables, $\boldsymbol{x} = [V_a\ \alpha\ \beta\ \phi\ \theta\ p\ q\ r]^{\intercal}$ Note that both models share the same Flight Envelope constraints and all of the involved variables appear in both of them.

However, for the calculation of a Trim FE, there are only 6 independent states. By choosing $[V_a\ \alpha\ \beta\ \phi\ \theta\ r]^{\intercal}$ as the independent state vector to describe a trim point and enforcing the coordinated turn equations, the other two states are derived by

$$k = r/\cos\phi/\cos\theta$$

$$p = -k\sin\theta \tag{20}$$

$$q = k\sin\phi\cos\theta \tag{21}$$

D. Implementation Details

The time horizon of the low-level controller was set at 0.4 s and 0.02 s time-step (20 step look-ahead). The high-level controller was configured at one order of magnitude slower, at 4 and 0.2 seconds equivalently.

With the intention of implementing the MPC in real-time in an embedded computer, we have encoded the two optimal problems with the ACADO framework [20], employing Multiple-Shooting SQP. The autogenerated, compiled controller code of the low-level controller would require 0.645 ms per call on an Intel Core i7-6500/2.50 GHz processor, while the translational dynamics controller required 3.91 ms. Both controllers had ample time to run in real-time.

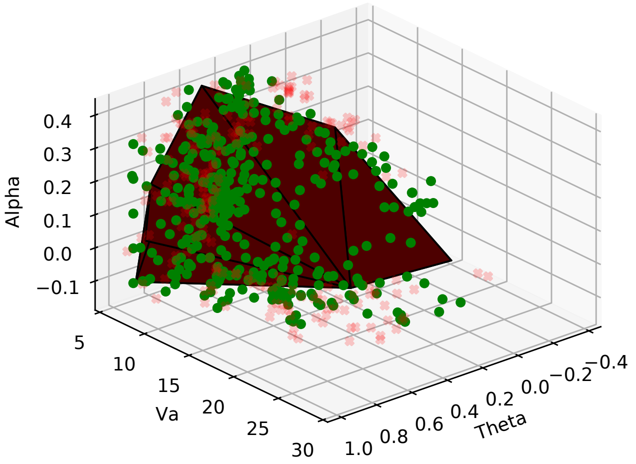


Fig. 3: A 3D slice of the final 6D Flight Envelope

TABLE I: Flight Envelope Determination Performance

	Search Dimension				
Operation	2	3	4	5	6
Trim Point Sampling Time (s)	0.25	0.83	1.16	3.72	17.80
Polytope Operations Time (s)	0.15	0.40	0.64	2.78	42.70

IV. Case Study

To demonstrate the capabilities of our proposed methodologies, we tested both the Flight Envelope determination algorithm and the MPC on a simulated fixed-wing UAV. The simulation model was of high detail and was written as a C++ shared library. A suitable Robotics Operating System (ROS) wrapper allowed the simulation to orchestrate the system simulation and control under a unified framework.

Regarding the proposed Flight Envelope determination algorithm, it was encoded in Python, interfacing with the C++ NLOPT library and the UAV model library for trim point sampling and the Parma Polyhedra Library (PPL) for certain polytope operations. A slice at the 3D space of an algorithm run can be seen in Figure 3.

Statistics on program performance can be seen in Table I. While for a smaller dimension of independent state variables the search is quite fast, at the full 6-dimensional space the procedure cannot be characterized as real-time, when initiated cold. However, it may be possible to perform FE update steps continuously at the maximum resolution level throughout and have a real-time FE description, adaptable to model variation. An appropriate real-time parameter identification algorithm would be needed in that case, such as in our previous work [21].

To assess the efficacy of the FE constraints in the MPC formulation, the following scenario was set up: Starting from level flight, the UAV was commanded to achieve an airspeed of $20\,\mathrm{m\,s^{-1}}$. This trim point was already at the limits of the airframe capabilities. Then the flight path angle reference was increased progressively in steps, pushing the reference point well outside of the FE. The results are depicted in Figure 4.

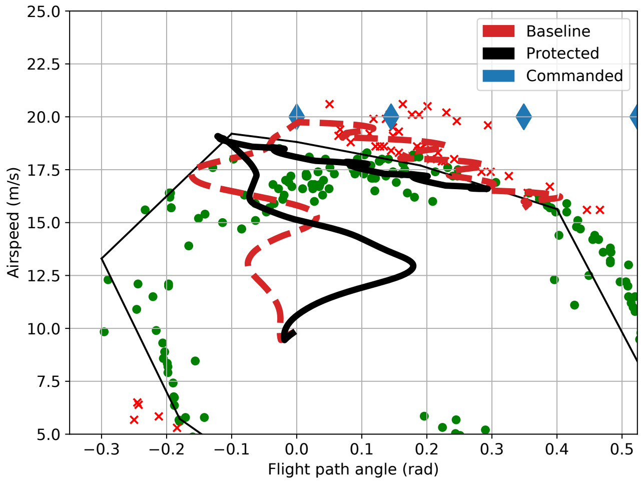


Fig. 4: UAV trajectories: Nominal MPC without FE constraints (dashed red), with FE constraints (black) and reference setpoints (blue diamonds).

The UAV trajectory governed by an MPC without FE constraints is marked by a dashed red line. Limited only by the airframe capabilities, the MPC pushses as hard as it can at the boundary of the FE, trying to achieve the setpoints, putting the UAV in danger of Loss-of-Control.

On the other hand, the response of the MPC under FE constraints can be seen in the solid black line. The upper bound of the FE limits the UAV trajectory, keeping it well inside the FE. Naturally this is at a cost of performance but in favour of safety: The UAV shouldn't have been issued a setpoint outside of its capabilities in the first place.

V. Conclusions and Future Work

In this work, we have presented a novel algorithm for approximate yet fast determination of the Trim Flight Envelope of a fixed-wing UAV, based on its general numerical model. This algorithm can potentially be run in real-time, since it is sequential in nature. With knowledge of the up-to-date aircraft parameters values (through online Parameter Identification), the FE constraints can be updated online, reflecting the maneuvering capabilities of the aircraft. Since aircraft parameters change in the presence of faults, the online FE can provide a safe operation boundary for the post-fault system.

In addition, a two-layer Model Predictive Controller (MPC) has been designed able to incorporate the Flight Envelope in the form of convex linear constraints. The overall control scheme has been encoded in fast real-time software and tested in simulation.

Two immediate future directions are meaningful to this work. First, evaluating the Flight Envelope determination in real-time, under the presence of faults, to investigate its behaviour during shifts in the Flight Envelope contour. Second, benchmarking the performance of the overall control system experimentally.

References

[1] M. R. Elgersma and B. G. Morton, "Nonlinear Six-Degree-of-Freedom Aircraft Trim," *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 2, pp. 305–311, mar 2000. [Online]. Available: http://arc.aiaa.org/doi/10.2514/2.4523

[2] M. G. Goman, A. V. Khramtsovsky, and E. N. Kolesnikov, “Evaluation of Aircraft Performance and Maneuverability by Computation of Attainable Equilibrium Sets,” *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 2, pp. 329–339, mar 2008. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/1.29336http://arc.aiaa.org/doi/10.2514/1.29336

[3] F. A. De Almeida, “Reference Management for Fault-Tolerant Model Predictive Control,” *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 1, pp. 44–56, jan 2011. [Online]. Available: http://arc.aiaa.org/doi/pdf/10.2514/1.50938http://arc.aiaa.org/doi/abs/10.2514/1.50938

[4] T. Keviczky and G. J. Balas, “Software-Enabled Receding Horizon Control for Autonomous Unmanned Aerial Vehicle Guidance,” *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 3, pp. 680–694, may 2006. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/1.15562

[5] J. Lygeros, “On reachability and minimum cost optimal control,” *Automatica*, vol. 40, no. 6, pp. 917–927, jun 2004. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0005109804000263

[6] E. R. van Oort, Q. P. Chu, and J. A. Mulder, “Maneuver Envelope Determination through Reachability Analysis,” in *Advances in Aerospace Guidance, Navigation and Control*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 91–102. [Online]. Available: http://link.springer.com/10.1007/978-3-642-19817-5{_}8

[7] T. Lombaerts, S. Schuet, K. Wheeler, D. M. Acosta, and J. Kaneshige, “Safe Maneuvering Envelope Estimation based on a Physical Approach,” in *AIAA Guidance, Navigation, and Control (GNC) Conference*. American Institute of Aeronautics and Astronautics, aug 2013, pp. 1–20. [Online]. Available: http://arc.aiaa.org/doi/10.2514/6.2013-4618

[8] J. Stapel, C. C. de Visser, E.-J. Van Kampen, and Q. P. Chu, “Efficient Methods for Flight Envelope Estimation through Reachability Analysis,” in *AIAA Guidance, Navigation, and Control Conference*, no. January. American Institute of Aeronautics and Astronautics, jan 2016. [Online]. Available: http://arc.aiaa.org/doi/10.2514/6.2016-0083

[9] S. Schuet, T. Lombaerts, D. Acosta, J. Kaneshige, K. Wheeler, and K. Shish, “Autonomous Flight Envelope Estimation for Loss-of-Control Prevention,” *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 4, pp. 847–862, apr 2017. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.G001729

[10] H. N. Nabi, T. Lombaerts, Y. Zhang, E. van Kampen, Q. P. Chu, and C. C. de Visser, “Effects of Structural Failure on the Safe Flight Envelope of Aircraft,” *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 6, pp. 1257–1275, jun 2018. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.G003184

[11] H. A. Izadi, B. W. Gordon, and Y. Zhang, “A data-driven fault tolerant model predictive control with fault identification,” in *2010 Conference on Control and Fault-Tolerant Systems (SysTol)*. IEEE, oct 2010, pp. 732–737. [Online]. Available: http://ieeexplore.ieee.org/document/5675981/

[12] F. A. De Almeida and D. Leißling, “Fault-Tolerant Model Predictive Control with Flight-Test Results,” *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 2, pp. 363–375, mar 2010. [Online]. Available: http://arc.aiaa.org/doi/10.2514/1.46108

[13] F. Bateman, H. Noura, and M. Ouladsine, “Fault Diagnosis and Fault-Tolerant Control Strategy for the Aerosonde UAV,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 3, pp. 2119–2137, jul 2011. [Online]. Available: http://ieeexplore.ieee.org/document/5937287/

[14] K. McDonough and I. Kolmanovsky, “Fast Computable Recoverable Sets and Their Use for Aircraft Loss-of-Control Handling,” *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 4, pp. 934–947, apr 2017. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.G001747

[15] N. M. Sirakov, “A New Active Convex Hull Model for Image Regions,” *Journal of Mathematical Imaging and Vision*, vol. 26, no. 3, pp. 309–325, dec 2006. [Online]. Available: http://link.springer.com/10.1007/s10851-006-9004-6

[16] P. Marquez-Neila, L. Baumela, and L. Alvarez, “A Morphological Approach to Curvature-Based Evolution of Curves and Surfaces,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 2–17, jan 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6529072/

[17] G. Takács, “Convex polyhedron learning and its applications,” Ph.D. dissertation, 2009. [Online]. Available: http://www.sze.hu/{∼}gtakacs/download/tg{_}phdthesis.pdf

[18] B. Stevens, F. Lewis, and E.N. Johnson, *Aircraft Control and Simulation*, 3rd ed. Wiley, 2016, no. 9.

[19] R. W. Beard and T. W. McLain, *Small Unmanned Aircraft: Theory and Practice*. Princeton University Press, 2012. [Online]. Available: http://books.google.com/books?id=ZR8PuVUy-ygC{&}pgis=1

[20] B. Houska, H. Ferreau, and M. Diehl, “ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization,” *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298–312, 2011.

[21] P. Vaiopoulos, G. Zogopoulos-Papaliakos, and K. J. Kyriakopoulos, “Online Aerodynamic Model Identification on Small Fixed-Wing UAVs with Uncertain Flight Data,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2018, pp. 6587–6592. [Online]. Available: https://ieeexplore.ieee.org/document/8460585/