A SYSTEMS ENGINEERING APPROACH TO COLLABORATIVE COORDINATION OF UAS IN THE NAS WITH SAFETY GUARANTEES

Jacob Moschler, Yuchen R. Zhou, and John S. Baras, Institute for Systems Research,
A. James Clark School of Engineering, University of Maryland, College Park
Jungwoo Joh, Department of Electrical and Computer Engineering,
University of Maryland, College Park

Abstract

We develop new fundamental methodologies for high performance and provably safe autonomous and collaborative control and operation of autonomous unmanned aerial systems (UAS) in the national airspace (NAS), where both UAS and manned aircraft fly. The proposed framework is model-based, and emphasizes multiple scales in time and space as well as hybrid systems mathematics to capture both the analog and logical components of control functionalities. We develop on-line control laws that allow for multiple UAS agents to reconfigure to different formations with proofs of safety and convergence while navigating in integrated airspace with piloted air vehicles. We demonstrate our results in simulated scenaria with both cooperative and uncooperative air vehicles.

Introduction

To address directly the issues associated with performance vs. safety in these environments we develop a modeling structure inspired by the very successful research on Boids. In Boids simulations and animations, realistic (life-like), complex, navigating obstacles to a goal, fast flight patterns/behaviors for groups of birds/vehicles have been demonstrated in real-time [1]. Our approach allows us to represent individual aircraft within the UAS as autonomous agents having multiple behavioral states (modes): taking off, landing, patrolling, intercepting, and so on. In each state there is active control, the UAV must steer, adjust its speed, kinematics and take other actions based on its perception of the local environment, but these will be tempered by its current behavioral states. We provide a methodology which can be used to build individual control algorithms for any state. We create a "patrol" state to demonstrate the methodology.

We build upon the basic framework, developed in our earlier work, for studying the collaborative control of unmanned vehicles in uncertain and adversarial environments, with emphasis on differentiating between the sensing range and actuation range of the vehicles required for obstacle avoidance, moving threat avoidance and target following. Encoding the local information-based goals of each agent as a potential function, we design paths by a distributed gradient descent algorithm.

Of central importance and emphasis in our approach is the development of principles for control and operation of UAS in crowded airspace while maintaining provable levels of safety. In our approach we begin by developing a modeling, simulation and analysis/synthesis environment for UAS operations in such challenging environments with modular models of sensors and inference and modular models of control at different levels of locality. Our models allow us to use a rich array of analytical and numerical schemes and tools for performance and safety evaluation. A hardware abstraction layer allows us to design algorithms useful for multiple platforms while using hardware-specific models and controllers for demonstration purposes.

We describe our results in the joint characterizations of performance and safety. We use a combination of multi-criteria optimization, planning/re-planning, constraint-based reasoning, formal safety methods (set valued dynamic programming for reachable states and path computations) and model checking methods for hybrid dynamical systems. A typical example is the planning of safe paths for UAS with limited information about the environment.

Our methods ensure that the flight paths (checked for obstacle avoidance) terminate in the reachability set of safe dynamic states when

978-1-4799-4891-8/14/$31.00 ©2014 IEEE

executing dynamically feasible, very agile maneuvers. Finally we address complexity and real-time issues associated with the algorithms developed.

Literature Review

A study was made of various tools and techniques in order to determine state of the art methods for handling the problem of aerial robotic search. The overall structure of our approach frames the problem as synergistic planning between multiple layers of abstraction, as imagined by Bhatia and Kavraki in their studies of motion planning with complex goals. This multi-layer synergistic approach is shown to improve computational time by an order of magnitude [2].

Approaches to Heuristic Path Planning

We investigated the use of various methods to create a heuristic algorithm for planning and re-planning.

Cellular decomposition is one method that has been used by many high and low level robotic search and patrol algorithms. Cellular decomposition of an underwater area has been shown to provide dynamic coverage by autonomous underwater vehicles [3]. Approximate cellular decomposition is similar to search methods used by insects in which traces of pheromones allow for areas to be efficiently explored and monitored [4]. Virtual corridors can be used in addition to, or in place of cells to cover a large search area. These are in some ways advantageous to decomposing the search area into cells because they allow each agent to retain some data about the known map. Dividing a search area into corridors also allows for detailed safety analysis. Constraint-based reasoning allows these corridors to be kept within the flight envelope of a particular aircraft by minimizing path characteristics such as snap (second derivative of acceleration) [5].

Previous approaches to underwater search have included the use of a multiresolution grid. When performing a local search, the robotic agent may have limited spatiotemporal information so it relies on a local navigation function. Global search can be performed using a different function that uses more abstract information about a larger area. Varying levels of local and global navigation can be managed using potential cost functions [6]. For example, after searching a local area for some time and finding nothing, each individual robot could gradually and automatically transition from its local navigation algorithm towards a global search function.

Current Sensing Capabilities

For path planning that incorporates traffic collision avoidance, fast-approaching large aircraft can be modeled as moving obstacles or threats to be avoided. Until reliable, small-scale sensors become available, GBSAA (Ground-Based Sense And Avoid) is a reliable method for detecting approaching aircraft [7]. COTS PCAS (Portable Collision Avoidance Systems) are also available and may be integrated with unmanned aircraft as a secondary method to obtain position information about large aircraft nearby with transponders broadcasting ADS-B (Automatic Dependent Surveillance-Broadcast) signals [8].

To sense targets on the ground, onboard local sensing packages can be used by multiple robots to sense and dynamically track targets. Such tracking methods have been thoroughly studied [9]. More advanced turnkey sensing and target tracking systems are also available [10].

Approaches to Low-Level Control

Decentralized path generation and collision avoidance can also be accomplished using potential cost functions. Gradient descent algorithms can give robots a tendency to attract towards a target while repelling obstacles and other robots [11]. Artificial potential fields have also been used to provide collision avoidance to robotic manipulators [12].

Many aspects of the low-level control are hardware specific. We chose to demonstrate our approach on the increasingly popular electric quadrotor hardware platform, which can be configured to carry a useful payload of up to 58 kg or configured to fly for up to 85 minutes [13].

Model Predictive Control has been extensively studied for quadrotor aircraft and shown to improve the Integral Absolute Derivative control signal index for all measured states of flight [14]. Waypoint control has also been successfully implemented for the quadrotor hardware, and static obstacles can be avoided even while maneuvering quickly [15].

Approaches to Formal Safety Verification

Given a potential gradient and some expectation of the future environment model, one can produce an estimation of the states of the aircraft and the environmental states using the underlying dynamics and expected controller inputs. This generates tubes around each agent of the UAS as well as larger, fast-moving aircraft following scheduled trajectories. To capture modeling uncertainty, sensor noise and unpredictable disturbances such as wind gusts, enlargement of the tubes to consider all uncertainties is essential. The safety verification process then requires computation of the locations of each tube and possible collisions between the tubes. The tubes are formally defined as reachable sets which can be numerically represented using various methods available in the hybrid system analysis community [16-24]. Reachability analysis for nonlinear systems such as biological systems and autonomous vehicles has been a main theme of recent hybrid systems research [22, 23, 25]. This analysis is increasingly popular because nonlinear systems can be captured using hybridization [25], and methods developed in hybrid system verification can be easily adapted to nonlinear systems [22, 23]. In this work, we extend the reachability analysis further for safety verification of aircraft.

Overview and Framework

Our approach is structured with a high-level search algorithm (we use heuristic path planning as an example), a low-level control algorithm, and a formal mathematical verification of the system's safety and ability to avoid collisions, as shown in Figure 1. This multilayer framework lends itself well to a wide variety of use-cases and robot tasks, and can be used to develop and analyze future algorithms. We propose this framework for our modeling environment which is modular in that a variety of algorithms, aircraft with varying flight envelopes, and sensor packages can be easily put together and tested.

In our initial example using heuristic path planning as the high-level search layer, the synergistic interface in Figure 1 is a key part of our design process that allows us to integrate high and low-level controllers while maintaining formal safety verification.

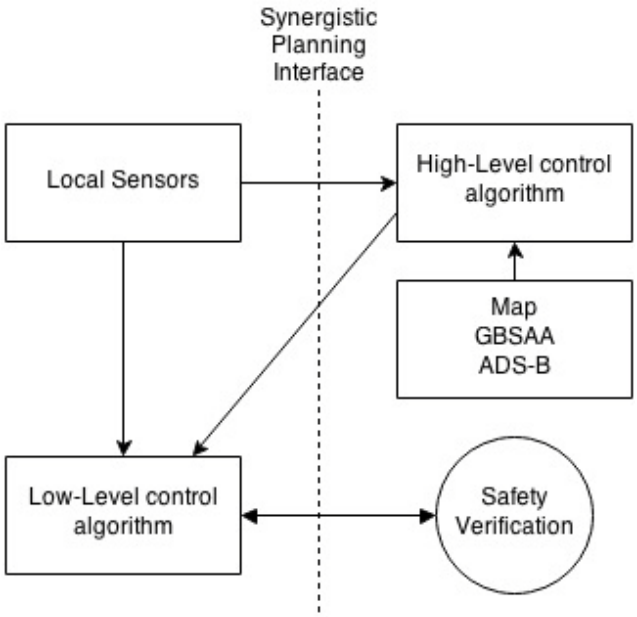

Figure 1. Multilayer Synergistic Planning

Heuristic Path Planning

We incorporate a heuristic path planning scheme as an example of a high-level algorithm to demonstrate our framework.

The following heuristic method is one example of how a simple shared memory requiring low communication bandwidth can be used to "oversee" a multi-agent search while keeping centralized control to a minimum and allowing the agents to adapt and re-configure if one agent becomes unable to perform its desired task.

The heuristic planning algorithm works as follows: A desired coverage area, in which agents of a UAS will explore to collect mapping data, contains stationary obstacles including control towers and hangars. Piloted aircraft taking off and landing are modeled as moving obstacles. While stationary obstacles may have fixed positions and shapes, the incoming aircraft are highly dynamic. Therefore the high-level heuristic planning of the UAS updates with each discreet time step. Between time steps, agents of the UAS explore the target coverage area with all stationary obstacles. After each time step, target coverage area changes reflecting new obstacle information. With the updated coverage area, each UAS updates its map.

Pseudo code is presented in Table 1 to describe the high-level heuristic algorithm.

Table 1. High Level Heuristic Planning

```
get target coverage area
get sensor range
set reference orientation: vertical or horizontal
set multiple UAS agents to arbitrary position
    in the known target coverage area

while
  if target coverage area is updated then
    do decompose the known target coverage
        area through reference orientation with
        defined width equal to sensor range
    set corridors along the decomposed lines
    set waypoints at the end points of corridors

    if UAS agents are out of corridor then
      move agents to the nearest corridor
    end
  end

  if two agents are approaching in the same
      corridor then
    set two waypoints in the middle of the
        corridor
  end

  snap each agent forward
  record sensor information

  if agent reaches a waypoint then
    move the agent to the next waypoint
  end

  if preset time is elapsed then
    update target coverage area
  end
end
```

To demonstrate, the high-level heuristic planning algorithm begins by accepting input about the target coverage area at an arbitrary time step. The range of onboard local sensors and the desired orientation of decomposition corridors must also be input by the user. To test the algorithm, we arbitrarily place agents of the UAS on the given target coverage area with obstacle positions fixed.

The algorithm then decomposes the target coverage area based on given sensor range and desired decomposition orientation. Separation of decomposition corridors is determined by the radius of sensor range. Agent-to agent collision avoidance redundancy is added by the low-level algorithm and verified by the safety function. Corridors are defined along the decomposed lines with waypoints at the end of each corridor [3, 5]. Waypoints are represented with an 'x' mark at the end of each columns in Figure 2.

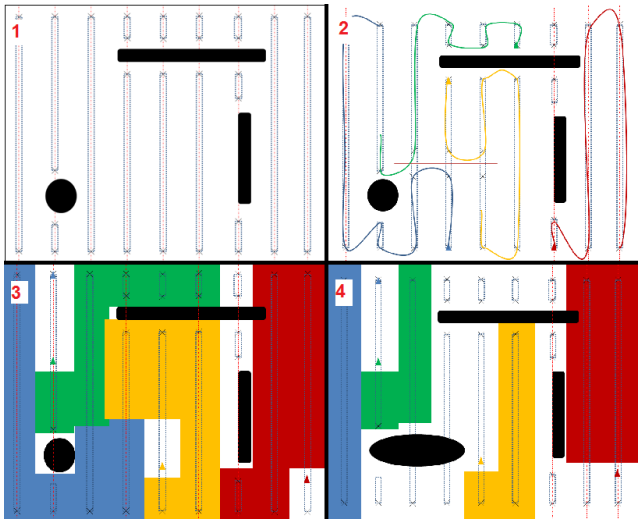

Figure 2. Heuristic Patrol Algorithm

Initially in the test case, arbitrarily deployed UAS are out of corridors in the target coverage area at a fixed time. The UAS agents automatically move to the nearest waypoints.

Because agents move to the nearest waypoints without any other requirements besides low-level collision avoidance, one corridor may end up with multiple agents. More waypoints are then set in the middle of the corridor. The newly set waypoints indicate to agents where to switch to other corridors so that efficient sensor coverage is achieved without redundant coverage.

When any agent reaches a preset waypoint, it then moves to the next waypoint to continue its patrol. Any time a corridor ends up having multiple agents, the algorithm once again sets new waypoints in the middle of the corridor.

The coverage area updates regularly to reflect latest obstacle information [5]. Decomposition is

repeated with the given sensor range and corridors are updated along the decomposed lines. New waypoints are set which may be different from the previous waypoint positions due to movement of dynamic obstacles.

After a corridor has been covered by the system, it will be assigned a coverage decay time. When the coverage area is updated, waypoints will be also created for areas whose coverage has decayed beyond the time limit. The decay times are shown by colored areas in Figure 2. This decay creates a continuous patrol state which the map continuously updates. If individual agents are reassigned, become disabled, require maintenance, refueling or recharging, remaining, those agents undergo a state change and are then governed by another algorithm. Remaining agents will distribute themselves to maintain coverage, and collisions are avoided by the low-level control algorithm which still runs on every agent. The coverage decay time may be adjusted by the user for any desired map refresh rate.

Low-Level Control

As described in the overview and Figure 1, our framework also employs a low-level control algorithm for basic target following and traffic collision avoidance.

We describe a hardware-agnostic low-level controller that consists of a minimized potential function. Since this function is implemented above a hardware abstraction layer, it can be applied to any aircraft hardware given that adequate dynamical models and MPC controllers exist for that hardware. The potential function receives information about flight envelope and velocity from separate, hardware-specific models. Taking a gradient descent approach then gives the potential controller a tendency to "repel" around static obstacles and "attract" towards a desired path given by the high-level heuristic path planner. This technique has also been shown to enable robust flocking and formation capabilities in decentralized systems [11].

This potential field method of robotic control may seem elegantly simple, however it has been shown to create a risk of the robot becoming trapped in local minima of the potential field [26]. Most prior results only cause convergence to local minima, so the potential field method has been difficult to use for global control of a robot swarm. Several options exist to address the problem of becoming trapped in local minima, making the potential field method more useful globally. Perhaps the most interesting solution to local minima trapping is the adaptation of methods previously used in image processing and computer vision to the control of robotic swarms. It has been shown that Gibbs Random Fields can be used to allow robot swarms to complete global objectives using only local interactions while maintaining the ability to avoid becoming trapped in local minima [27]. A useful tool was also created specifically to address the local minima problem in robotic swarm control using Markov Random Fields, building on the aforementioned Gibbs Random Fields method [28]. These recent solutions are promising ideas for using the potential field method to navigate a global map using only local data. Once modular simulations are integrated as we have proposed, our system will be tested to use the potential method globally for our UAS, possibly eliminating the need for the the high-level heuristic method described in the earlier section.

Simplistic use of the potential field method will be attempted first, using input from the high-level heuristic to keep the UAS agents near only desired local minima and keep them from becoming trapped. While not elegant, this simplistic method is sufficient since it only requires convergence to local minima, which most previous approaches using the potential field method have allowed for as long as initial positioning is close enough to the desired position [29, 30]. Global objectives are handled by our high-level heuristic algorithm.

Potential Function

The potential function is designed so that agents could be "pushed" away from obstacles and the paths of large aircraft using virtual potential fields while being "attracted" toward its desired path, received from a high-level controller. The potential function of every agent has multiple terms to give the agent input on how to deal with various types of obstacles and targets.

$$J_{\mathrm{a}}(\mathbf{x}_{\mathrm{a}}) = \sum_{i=1}^{N_{\mathrm{a}}} b_i f(\mathbf{x}_{\mathrm{a}}, \mathbf{x}_{\mathrm{a}i}, \mathbf{v}_{\mathrm{r}i})^{-1} + g(\mathbf{x}_{\mathrm{a}}, \mathbf{x}_m, \mathbf{v}_m)^{-1} + \left\| \mathbf{x}_{\mathrm{a}} - \mathbf{x}_{\gamma \mathrm{a}} \right\|_2^2$$

$$\mathbf{x}_{\mathrm{a}} \in \mathbb{R}^3 \setminus \mathcal{O}$$

$\mathbf{x}_{\mathrm{a}}$ presents the agent position within 3D position

space except the obstacle space $\mathcal{O}$. The first term of the potential function is

$$\sum_{i=1}^{N_a} b_i f(\mathbf{x}_a, \mathbf{x}_{ai}, \mathbf{v}_{ri})^{-1} .$$

In this expression, a special function, f is used to shape virtual potential fields around other aircraft in the collaborative swarm. b_i is a Boolean variable representing whether the i^{th} agent is close to the host agent a. N_a is the total number of collaborative agents.

For the inputs:

$$\text{Agent position} : \mathbf{x}_a$$

$$\text{Position of } i\text{th agent} : \mathbf{x}_{ai}$$

$$\text{Relative velocity towards } i\text{th agent} : \mathbf{v}_{ri}$$

$$f(\mathbf{x}_a, \mathbf{x}_{ai}, \mathbf{v}_{ri}) = (\mathbf{x}_a - \mathbf{x}_{ai})^T P_{ai} D_{ai} P_{ai}^T (\mathbf{x}_a - \mathbf{x}_{ai})$$

where

$$P_{ai} = \begin{pmatrix} \frac{v_{rix}}{\|\mathbf{v}_{ri}\|} & & \\ \frac{v_{riy}}{\|\mathbf{v}_{ri}\|} & \vec{o}_1 & \vec{o}_2 \\ \frac{v_{riz}}{\|\mathbf{v}_{ri}\|} & & \end{pmatrix},$$

$$D_{ai} = \begin{pmatrix} \|\mathbf{v}_{ri}\| & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

where $\vec{o}_1, \vec{o}_2 \in \mathbb{R}^3$ are picked to make the columns of the matrix P_{ai} orthonormal. This function f is similar to a squared Euclidean norm, but is modified to incorporate relative velocity. This term pushes the agent away from nearby agents, thus the potential function is inversely proportional to the value of $f(\mathbf{x}_a, \mathbf{x}_{ai}, \mathbf{v}_{ri})$. This function serves to capture position and velocity information about nearby UAS agents and create individual potential fields based on the relative velocity between the agent and each of its collaborators within a given communication radius.

In the second term of the potential function,

$$g(\mathbf{x}_a, \mathbf{x}_m, \mathbf{v}_m)^{-1} = \left((\mathbf{x}_a - \mathbf{x}_m)^T P_m D_m P_m^T (\mathbf{x}_a - \mathbf{x}_m)\right)^{-1}$$

a similar function, g is used to shape virtual potential fields around the trajectories of piloted aircraft. P_m and D_m are constructed similar to P_{ai} and D_{ai} but with $\mathbf{v}_{ri}$ changing to relative velocity $\mathbf{v}_m$ of the piloted aircraft and the host agent. This function, g is designed to create the steepest gradient of all the terms in the potential control algorithm. As a result, the unmanned system will always have the strongest of its virtual potential fields pushing UAS agents out of the way of any piloted aircraft. As described in the literature review, the positions and velocities of incoming piloted aircraft might be determined via GBSAA or by an onboard system that interfaces with ADS-B.

The final term of the potential function,

$$\left\|\mathbf{x}_a - \mathbf{x}_{\gamma a}\right\|_2^2$$

serves to attract each agent to the trajectory it has been assigned by the high-level control algorithm. The agent is attracted to a moving target point $\mathbf{x}_{\gamma a}$ forward of it along the path γ_a, which presents itself as a moving minima that will be "chased" by the UAS agent as it moves around. An algorithm was created to cause this target to move along the path and, when necessary, wait for the UAS agent. This algorithm takes as its input the path from the high-level heuristic algorithm. The target moves along the path while the UAS is within a given range. If the UAS falls too far behind the target, the target stops moving until the UAS agent is once again within range. Thus if the agent is slowed down for any reason (perhaps waiting for a piloted aircraft to pass, as dictated by previous terms in this function) then the target point will stop and "wait," only to resume moving on the programmed path when the agent is within a given distance from the target point.

Summing these potential terms creates the theoretical dynamic surface, visualized in Figures 3 and 4. Descending the gradient of this sum gives the controller a tendency to "roll downhill," steering agents of the UAS towards their moving target and around collisions.

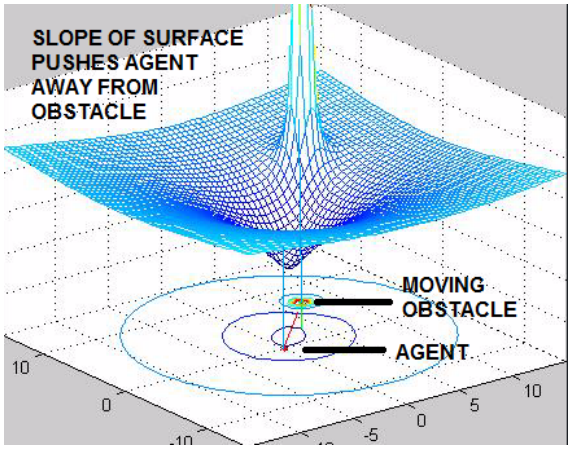

Figure 3. Visualization of Dynamic Surface

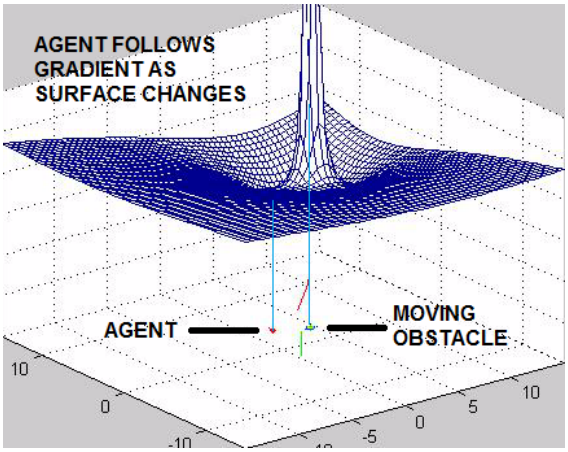

Figure 4. Visualization of Dynamic Surface

Dynamical Model

Low-level control methods such as model-predictive control require an accurate dynamical model of the hardware platform. In our example, we use a dynamical model of the AscTec Hummingbird aircraft that has been validated through experimental testing [31, 32]. This model has a 12-dimensional state variable **x** and control input **u** based on thrust and rotational momentum. [4] The components of **x** are defined by the following,

$$\mathbf{x} = (p_x, p_y, p_z, v_x, v_y, v_z, \phi, \theta, \psi, p, q, r),$$

where $\mathbf{p} = (p_x, p_y, p_z)$ is a position vector in the global coordinate frame, $\mathbf{v} = (v_x, v_y, v_z)$ is a velocity vector in the global coordinate frame, attitude variables are (ϕ, θ, ψ) and body frame rotational velocity is $\boldsymbol{\Omega} = (p, q, r)$.

The control inputs are described by a four dimensional input variable $\mathbf{u} = (F, M_1, M_2, M_3)$, where F is the total thrust along the centerline of the body frame, and $\mathbf{M} = (M_1, M_2, M_3)$ is the rotational momentum.

The overall dynamics can then be captured by the following vector differential equations,

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{v} \\ \dot{\mathbf{v}} = g\mathbf{e}_3 - \dfrac{F}{m} R\mathbf{e}_3 \\ \dot{\mathbf{R}} = R_w^{-1}\boldsymbol{\Omega} \\ \dot{\boldsymbol{\Omega}} = J^{-1}(\mathbf{M} - \boldsymbol{\Omega} \times J\boldsymbol{\Omega}) \end{cases}$$

$$R = \begin{pmatrix} \mathrm{c}\theta\mathrm{c}\psi - \mathrm{s}\theta\mathrm{s}\phi\mathrm{s}\psi & -\mathrm{c}\phi\mathrm{s}\psi & \mathrm{c}\psi\mathrm{s}\theta + \mathrm{c}\theta\mathrm{s}\phi\mathrm{s}\psi \\ \mathrm{c}\psi\mathrm{s}\theta\mathrm{s}\phi + \mathrm{c}\theta\mathrm{s}\psi & \mathrm{c}\phi\mathrm{c}\psi & -\mathrm{c}\theta\mathrm{c}\psi\mathrm{s}\phi + \mathrm{s}\theta\mathrm{s}\psi \\ -\mathrm{c}\phi\mathrm{s}\theta & \mathrm{s}\phi & \mathrm{c}\theta\mathrm{c}\phi \end{pmatrix}$$

$$R_w = \begin{pmatrix} \mathrm{c}\theta & 0 & \mathrm{c}\phi\mathrm{s}\theta \\ 0 & 1 & \mathrm{s}\phi \\ \mathrm{s}\theta & 0 & \mathrm{c}\phi\mathrm{c}\theta \end{pmatrix}$$

where $\mathbf{e}_3 = [0, 0, 1]^{\mathrm{T}}$.

A very nice property of this dynamic or helicopter dynamics is differential flatness. In other words, system states and control inputs can be written as algebraic functions of carefully selected flat outputs and their derivatives. For quadrotor dynamics in particular, it is proved in [4], that the flat outputs can be $\mathbf{y} = (p_x, p_y, p_z, \psi)$. It also has been proved that a simple linear controller in the form of equation (1) provides asymptotic stability to the reference trajectory $\mathbf{y}_d$.

$$\begin{aligned} \mathbf{u} &= -K_p(\mathbf{y} - \mathbf{y}_d) - K_d(\dot{\mathbf{y}} - \dot{\mathbf{y}}_d) \\ &= -K(\mathbf{x} - \mathbf{x}_d) \end{aligned} \qquad (1)$$

where K can only have nonzero values corresponding to the flat outputs and their derivatives.

This dynamical model and controller of the hardware platform can be easily adjusted using information from existing hardware databases to model other quadrotor platforms. Other models can be used for standard rotorcraft and fixed-wing aircraft, although the absence of this differential

flatness property could cause analysis of those models to be more complex.

Formal Safety Verification

Previous sections address safety control and path planning, during the real time execution. But many situations can occur that are either not expected due to mathematical assumptions and definition of the scope, or environmental factors that are not well captured in the model such as wind gusts and error in sensing position. Safety verification is another layer to improve the safe operation of the system and ensure the safety requirements are met deterministically and stochastically. Safety verification can be done using temporal logic [33], barrier certificates [34], theorem proving [35] and reachability analysis [16-24, 36]. In this work we emphasize reachability analysis, which formally determines the reachable regions of state variables such as position, velocity, and aircraft attitude over time. Systems that can be analyzed using reachability analysis can be as simple as a linear time invariant (LTI) systems and as complicated as hybrid systems with nonlinear dynamics. The reachability analysis to which we are referring here is based on multiple references [18-25, 37]. Comparing the reachability analysis and optimal control synthesis method proposed in [38-41], the former works have much better real-time computational capabilities. In this work, the reference trajectory input is assumed to have very high fidelity and optimality in the large scale. The safety verification is used as an extra layer of protection against uncertainty and disturbance in the system. If the reference trajectory is unsafe, the system is then required to provide a new trajectory that is biased to a safer region. The simplest dynamics for reachability analysis are performed for linear time invariant (LTI) systems [18, 20].

Linear Time Invariant System

Reachability analysis for linear time invariant systems (LTI) is well developed. The LTI system we are referring to is described by the following linear ordinary differential equations (ODE) with initial conditions.

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \mathbf{u}(t) \quad \mathbf{x}, \mathbf{u} \in \mathbb{R}^n, \mathbf{x}(0) = \mathbf{x}_0$$

The solution of the ODE with fixed initial conditions is the following,

$$\mathbf{x}(t) = e^{tA}\mathbf{x}_0 + \int_0^t e^{(t-\tau)A}\mathbf{u}(\tau)d\tau$$

The idea of computing the reachable set is based on the following facts regarding the solution. The first term is related to the initial condition but not the control input, while the second term captures the state dynamics due to the control inputs. This separation allows parallel computation of the reachable set and computation of the set due to initial condition and inputs, and dramatically improves the computational efficiency.

The associated LTI reachable set computation is setup as follows,

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \mathbf{u}(t), \quad \mathbf{x}(0) \in \mathcal{X}_0, \mathbf{u}(t) \in \mathcal{U}$$

where $\mathcal{X}_0$ and $\mathcal{U}$ represent the initial set and control set. The set $\mathcal{U}$ also captures the noise and disturbances of the system. For general time varying linear systems, the setup is similar [42].

Because the solution of the ODE is a sum of a term solely depending on the input terms and a term depending on the initial condition set, the reachable set $\mathcal{R}_\delta(\mathcal{X}_0)$, within a δ time horizon can be described by the following,

$$\mathcal{R}_\delta(\mathcal{X}_0) = e^{\delta A}\mathcal{X}_0 \oplus \mathcal{R}_\delta(\{0\})$$

Effective reachable set representations for large dimensional systems such as aircraft include support functions [19-20], polytopes [17, 43] and zonotopes [21-24, 42, 44]. We use the zonotopes representation, because zonotopes perform well for linear systems [44]. The tool Continuous Reachability Analyzer (CORA), which implements zonotope-based reachability analysis, runs in the Matlab environment. It has been implemented with the capability of verifying the safety of highly nonlinear autonomous vehicles traversing a reference trajectory [21-23].

Nonlinear System

The nonlinear system studied here is described by the following general ODE,

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}, \mathbf{u}) \quad \mathbf{x}(0) \in \mathcal{X}_0, \mathbf{u}(t) \in \mathcal{U}$$

Most reachability analysis of nonlinear systems is performed by abstracting to differential inclusions of simpler dynamics such as hybrid systems with linear dynamics [25], or linearized around reference trajectories [21-23]. The latter outperforms partition-based methods because partition-based methods are commonly fixed. Approaches [38-41], which do not use abstraction, are commonly based upon optimization techniques to solve the Hamilton-Jacobi-Isaacs (HJI) partial differential equation (PDE), which does not scale well for real time large dimensional systems. Abstractions using linearization are also useful for dealing with very complex systems such as helicopters and high-order aircraft models [22].

A simple linearization can be described as follows,

$$\begin{aligned}\dot{\mathbf{x}}(t) &= f(\mathbf{x},\mathbf{u},\mathbf{x}_d)\\ &= f(\mathbf{x}^*,\mathbf{u}^*,\mathbf{x}_d) + \nabla_{\mathbf{x}} f\big|_{\mathbf{x}=\mathbf{x}^*}(\mathbf{x}-\mathbf{x}^*)(t)\\ &\quad + \nabla_{\mathbf{u}} f\big|_{\mathbf{u}=\mathbf{u}^*}(\mathbf{u}-\mathbf{u}^*)(t) + \text{higher order term},\\ \mathbf{x}(0) &\in \mathcal{X}_0,\ \mathbf{u}(t) \in \mathcal{U}\end{aligned}$$

Furthermore, the nonlinear dynamics can be differential, included by the simplified linear dynamics if the Lagrangian remainder term of the Taylor series expansion is described carefully. Let $\mathbf{z} = [\mathbf{x},\mathbf{u}]$, then the inclusion can be formally described as,

$$\begin{aligned}\dot{\mathbf{x}}(t) \in\ & f(\mathbf{z}^*,\mathbf{x}_d) + \nabla_{\mathbf{z}} f\big|_{\mathbf{z}=\mathbf{z}^*}(\mathbf{z}-\mathbf{z}^*)(t)\\ & + \frac{1}{2}(\mathbf{z}-\mathbf{z}^*)^T \nabla_{\mathbf{z}}^2 f\Big|_{\mathbf{z}\xi}(\mathbf{z}-\mathbf{z}^*)(t)\end{aligned} \tag{2}$$

The ξ in equation (2) takes values in the set. $\left\{\mathbf{z}^* + \alpha(\mathbf{z}-\mathbf{z}^*)\,\middle|\,\alpha \in [0,1]\right\}$.

Quadrotor Dynamics with Proportional and Derivative (PD) Trajectory Tracking Controller

Because of the differential flatness property of the commonly used quadrotor dynamics, the trajectory tracking controller can be a simple PD controller described in equation (1).

Letting $\mathbf{z} = [\mathbf{x},\mathbf{u}]$, the overall dynamics of the quadrotor then becomes the following,

$$\begin{aligned}\dot{\mathbf{x}}(t) \in\ & f(\mathbf{z}^*,\mathbf{x}_d) + \nabla_{\mathbf{z}} f\big|_{\mathbf{z}=\mathbf{z}^*}(\mathbf{z}-\mathbf{z}^*)(t)\\ & + \frac{1}{2}(\mathbf{z}-\mathbf{z}^*)^T \nabla_{\mathbf{z}}^2 f\Big|_{\mathbf{z}\xi}(\mathbf{z}-\mathbf{z}^*)(t),\\ \mathbf{x}(0) &\in \mathcal{X}_0,\ \mathbf{u}(t) \in \mathcal{U}\end{aligned} \tag{3}$$

The set $\mathcal{U}$ in equation (3) is not control input set, but rather a disturbance set, which captures model uncertainty, sensor noise, and disturbances such as wind gusts which are not considered during the design and analysis of the PD trajectory tracking controller [4].

Computation using such methods has been demonstrated for real time applications of autonomous vehicles with high degrees of nonlinearity [21, 23]. The expected outcome of the reachability analysis includes reachable sets of position and orientation of autonomous aircraft along with piloted aircraft that are inside the surveying workspace. When there are nearby friendly aircraft which are autonomously controlled by the same high level control system, the reachability set data are shared and safety is verified by checking that the sets are not colliding in any instant of the time during the controller sampling period. The UAS operator will be able to obtain the reachable tubes of the autonomous aircraft in almost real time and transmit location and trajectory data to ATC and nearby aircraft if desired. Piloted aircraft always have the highest priority in our system, and autonomous aircrafts, in this case quadrotors, can reroute based on the GBSAA data, ADS-B transponder data, and onboard sensing data if equipped. If reachable sets collide, the high-level trajectory can be changed and the low level controller will be switched to an emergency avoidance state to use maximum power to move clear of the paths of piloted aircraft.

Conclusion

We were able to use the proposed framework along with modular models to develop a hybrid UAS control system that lends itself to multi-criteria optimization while assuring safety of flight using reachability analysis.

To test our framework, we developed and simulated a modular "patrol" use case where autonomous quadrotor type aircraft are programmed to complete a task in the vicinity of fast-moving

piloted aircraft, successfully avoiding their paths. Takeoff, intercept, and emergency behavioral states can also be defined using the aforementioned methodology and each state allows verification that UAS agents are kept safely away from the paths of piloted aircraft.

Future Work

After our methodology is used to create control schemes for each a variety of states, the states will be integrated into one system. Transitions from one state to another will be triggered by changes in each agent's internal/external environment. We will represent this concept as a timed finite state machine (TFSM). As a result the dynamical systems in our formulations will frequently be hybrid dynamical systems. Using temporal logic along with MPC - based approaches, we will address the low level control for real vehicles' motion and determine state changes based on a receding horizon.

Future work will https://www.youtube.com/watch?v=VpW_B9NHwaQalso include the use of approximate dynamic programming and Q-learning to implement multi-resolution search to create a search safe algorithm that can handle more complicated coverage problems.

References

[1] Reynolds, Craig W, 1987, Flocks, herds and schools: A distributed behavioral model, in ACM SIGGRAPH Computer Graphics, vol. 21, no. 4, ACM, pp. 25-34.

[2] Bhatia, Amit, Matthew Maly, Lydia Kavraki, and Moshe Vardi, 2011, Motion Planning with Complex Goals, IEEE Robotics & Automation Magazine, vol. 18, no. 3, pp. 55–64.

[3] Hert, Susan, Sanjay Tiwari, and Vladimir Lumelsky, 1996, A Terrain-Covering Algorithm for an AUV, in Underwater Robots, Springer, pp. 17–45.

[4] Choset, Howie, 2001, Coverage for robotics–A Survey of Recent Results, Annals of Mathematics and Artificial Intelligence, vol. 31, no. 1–4, pp. 113–126.

[5] Mellinger, Daniel, and Vijay Kumar, 2011, Minimum Snap Trajectory Generation and Control for Quadrotors, in 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 2520–2525.

[6] Gupta, Shalabh, James Hare, and Shengli Zhou, 2012, Cooperative Coverage Using Autonomous Underwater Vehicles in Unknown Environments, in Oceans, IEEE, pp. 1–5.

[7] Drake, P. R., and Robert J. Stamm, 2011, GBSAA Radar with Altitude Processing Supporting UAS in the NAS, in 2011 Integrated Communications, Navigation and Surveilance Conference (ICNS), IEEE, pp. 1–12.

[8] GDL 39 Portable ADS-B and GPS Receiver User's Guide, Garmin.

[9] Parker, Lynne E, 2002, Distributed Algorithms for Multi-Robot Observation of Multiple Moving Targets, Autonomous Robots, vol. 12, no. 3, pp. 231–255.

[10] Thermal Imaging Cores and Components, FLIR. http://www.flir.com/cs/emea/en/view/?id=42103

[11] Hovareshti, Pedram, 2009, Consensus Problems and the Effects of Graph Topology in Collaborative Control. PhD Thesis, University of Maryland. http://drum.lib.umd.edu/handle/1903/9304

[12] Khatib, Oussama, 1986, Real-Time Obstacle Avoidance for Manipulators and Mobile Robots, The International Journal of Robotics Research, vol. 5, no. 1, pp. 90–98.

[13] Frommann, Olaf. 2013, HK Beerlift 58,7kg 720p [Video file]. Retrieved from https://www.youtube.com/watch?v=VpW_B9NHwaQ.

[14] Raffo, Guilherme V., Manuel G. Ortega, and Francisco R. Rubio, 2010, An Integral Predictive/Nonlinear H_∞ Control Structure for a Quadrotor Helicopter. Automatica vol. 46, no. 1, pp 29-39.

[15] Richter, Charles, Adam Bry, and Nicholas Roy, 2013, Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments. http://www.michigancmes.org/papers/roy7.pdf

[16] Stursberg, Olaf, and Bruce H. Krogh, 2003, Efficient Representation and Computation of Reachable Sets for Hybrid Systems, in Hybrid

Systems: Computation and Control, Springer, pp. 482–497.

[17] Asarin, Eugene, Olivier Bournez, Thao Dang, and Oded Maler, 2000, Approximate Reachability Analysis of Piecewise-Linear Dynamical Systems, in Hybrid Systems: Computation and Control, Springer, pp. 20–31.

[18] Girard, Antoine, Colas Le Guernic, and Oded Maler, 2006, Efficient Computation of Reachable Sets of Linear Time-Invariant Systems with Inputs, in Hybrid Systems: Computation and Control, Springer, pp. 257–271.

[19] Le Guernic, Colas, and Antoine Girard. 2009, Reachability Analysis of Hybrid Systems Using Support Functions, in Computer Aided Verification, pp. 540–554.

[20] Le Guernic, Colas, and Antoine Girard, 2010, Reachability Analysis of Linear Systems Using Support Functions, Nonlinear Analysis: Hybrid Systems, vol. 4, no. 2, pp. 250–262.

[21] Althoff, Matthias, Daniel Althoff, Dirk Wollherr, and Martin Buss, 2010, Safety Verification of Autonomous Vehicles for Coordinated Evasive Maneuvers, in 2010 IEEE Intelligent Vehicles Symposium (IV), IEEE, pp. 1078–1083.

[22] Althoff, M., and J.M. Dolan, 2011, Set-Based Computation of Vehicle Behaviors for the Online Verification of Autonomous Vehicles, in 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC), IEEE, pp. 1162–1167.

[23] Althoff, Matthias, and John M. Dolan, 2012, Reachability Computation of Low-Order Models for the Safety Verification of High-Order Road Vehicle Models, in American Control Conference (ACC), IEEE, pp. 3559–3566.

[24] Althoff, Matthias, 2013, Reachability Analysis of Nonlinear Systems Using Conservative Polynomialization and Non-Convex Sets, in Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control, ACM, pp. 173–182.

[25] Dang, Thao, Colas Le Guernic, and Oded Maler, 2011, Computing Reachable States for Nonlinear Biological Models, Theoretical Computer Science, vol. 412, no. 21, pp. 2095–2107.

[26] Koren, Yoram, and Johann Borenstein. "Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation." Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on. IEEE, 1991.

[27] Baras, John S., Xiaobo Tan, 2004, Control of Autonomous Swarms Using Gibbs Sampling. In: Proceedings of the 43rd IEEE Conference on Decision and Control. Atlantis, Paradise Island, Bahamas (pp. 4752–4757).

[28] Xi, Wei, Xiaobo Tan, and John S. Baras, 2006, Gibbs Sampler-Based Coordination of Autonomous Swarms. Automatica 42.7 (2006): 1107-1119.

[29] Baras, John S., Xiaobo Tan, and Pedram Hovareshti, 2003, Decentralized Control of Autonomous Vehicles. Proceedings 42nd IEEE Conference on Decision and Control, Vol. 2. IEEE, 2003, pp. 1532-1537.

[30] Tan, Xiaobo, Wei Xi, and John S. Baras, 2010, Decentralized Coordination of Autonomous Swarms Using Parallel Gibbs Sampling. Automatica 46.12 (2010): 2068-2076.

[31] Mellinger, Daniel, Nathan Michael, and Vijay Kumar, 2012, Trajectory Generation and Control for Precise Aggressive Maneuvers with Quadrotors. The International Journal of Robotics Research, vol. 31, no. 5, pp. 664–674.

[32] Michael, Nathan, Daniel Mellinger, Quentin Lindsey, and Vijay Kumar, 2010, The GRASP Multiple Micro-UAV Testbed. IEEE Robotics & Automation Magazine, vol. 17, no. 3, pp. 56–65.

[33] Manna, Zohar, 1995, Temporal Verification of Reactive Systems: Safety, vol. 2. Springer, 1995.

[34] Prajna, Stephen, 2006, Barrier Certificates for Nonlinear Model Validation,. Automatica, vol. 42, no. 1, pp. 117–126.

[35] Platzer, André. 2010, Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics. Springer.

[36]Asarin, Eugene, Thao Dang, Goran Frehse, Antoine Girard, Colas Le Guernic, and Oded Maler, 2006, Recent Progress in Continuous and Hybrid Reachability Analysis. In 2006 IEEE Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications,

2006 IEEE International Symposium on Intelligent Control, IEEE, pp. 1582–1587.

[37] Frehse, Goran, Rajat Kateja, and Colas Le Guernic, 2013, Flowpipe Approximation and Clustering in Space-Time. In Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control, ACM, pp. 203–212.

[38] Tomlin, C.J., I. Mitchell, A.M. Bayen, and M. Oishi, 2003, Computational Techniques for the Verification of Hybrid Systems. In Proceedings of the IEEE, vol. 91, no. 7, pp. 986–1001.

[39] Mitchell, I.M., A.M. Bayen, and C.J. Tomlin, 2005, A Time-Dependent Hamilton-Jacobi Formulation of Reachable Sets for Continuous Dynamic Games. IEEE Transactions on Automatic Control, vol. 50, no. 7, pp. 947–957.

[40] Oishi, Meeko, Ian Mitchell, Claire Tomlin, and Patrick Saint-Pierre, 2006, Computing Viable Sets and Reachable Sets to Design Feedback Linearizing Control Laws under Saturation. In 2006 45th IEEE Conference on Decision and Control, IEEE, pp. 3801–3807.

[41] Hoffmann, Gabriel M., Steven L. Waslander, and Claire J. Tomlin, 2008, Quadrotor Helicopter Trajectory Tracking Control. In AIAA Guidance, Navigation and Control Conference and Exhibit, Honolulu, Hawaii, pp. 1–14.

[42] Althoff, Matthias, 2010, Reachability Analysis and Its Application to the Safety Assessment of Autonomous Cars. PhD Thesis, Technical University of Munich, Feb. 2010.

[43] Chutinan, Alongkrit, and Bruce H. Krogh, 2003, Computational Techniques for Hybrid System Verification. IEEE Transactions on Automatic Control, vol. 48, no. 1, pp. 64–75.

[44] Girard, Antoine, 2005, Reachability of Uncertain Linear Systems Using Zonotopes. In Hybrid Systems: Computation and Control, Springer, pp. 291–305.

Acknowledgments

Research partially supported by grants NSF CNS-1035655, NIST 70NANB11H148 and AFOSR MURI FA9550-09-1-0538.

Thanks also to Daniel Mellinger, Matthew Turpin, Kmel Robotics and Vijay Kumar's GRASP Laboratory at the University of Pennsylvania for helpful discussions of quadrotor dynamical models.

Email Addresses

John S. Baras: baras@umd.edu

Yuchen R. Zhou: yzh89@umd.edu

Jacob Moschler: jmoschle@umd.edu

Jungwoo Joh: jwjoh@umd.edu

2014 Integrated Communications Navigation and Surveillance (ICNS) Conference

April 8-10, 2014