Cooperative Formation Flying in Autonomous Unmanned Air Systems with Application to Training

Hongliang Yuan[1], Vivian Gottesman[2], Mark Falash[2], Zhihua Qu[1], Etyan Pollak[2], and Jiangmin Chunyu[1]

[1] School of EECS, University of Central Florida, Orlando FL 32826
[2] L-3 Communications, Link Simulation and Training, Strategic Technologies, Orlando, FL 32826

Abstract. The study of unmanned aerial systems (UAS) has been an active research topic in recent years due to the rapid growth of UAS real-world applications driven by the Global War on Terrorism (GWOT). UAS are defined as a complete unmanned system including control station, data links, and vehicle. Unmanned aerial vehicle (UAV) refers to the vehicle element of the UAS. Currently UAS operate standalone, independent of neighboring UAS and used primarily for reconnaissance. However UAS roles are expanding to the point where UAV swarms will operate as cooperative autonomous units. The reason is that cooperatively controlled multiple UAS have the potential to complete mission critical complicated tasks with the higher efficiency and failure tolerance, such as coordinated navigation to a target, coordinated terrain exploration and search and rescue operations.

This chapter presents study results associated with real-time trajectory planning and cooperative formation flying algorithms for use in performing multi-UAV cooperative operations. Closed form analytical and simulation results were used along with a UAS simulation test bed for evaluating and verifying these algorithms in multi-UAV cooperative scenarios. The full kinematics constraints of the UAV model is explicitly used, ensuring the planned trajectories and formations are feasible. Two operational modes are implemented for every UAV, one corresponding to the search phase, the other corresponding to the cooperative flying phase. Each phase is executed upon receiving commands. Finally this chapter discusses the use of this simulation environment for multi-UAV cooperative operator training.

1 Introduction

In order to provide a comprehensive solution for the trajectory planning problem, it should be recognized that the motion-planning of robots is analogous to the real-time trajectory planning of a group of UAV. Therefore, leveraging past research efforts devoted to the motion-planning problem of robots is directly applicable. Some popular approaches among them are potential fields, splines, and numerical methods such as the D* and A* search algorithm.

For the potential fields approach in [1] and [2], the trajectories are expelled away from obstacles by pre-built repulsive potential fields around the obstacles, and the goal is surrounded by an attractive potential field. This approach generally has multiple local minima and requires massive computational resources when applied to 3D applications.

To illustrate this, consider the repulsive potential field:

$$U(r) = \frac{1}{r^2}.$$

The attractive potential field is defined as

$$U(r') = r'^2,$$

where r, r' are the corresponding distances. A robot is to reach its goal along the gradient direction of its overall potential, that is,

$$U(r, r') = U(r) + U(r') = \frac{1}{r^2} + r'^2.$$

This scalar field has local minima close to the goal point. If a robot approches a local minima, it will become stuck. When multiple obstacles are injected into the scenario, the potential becomes more complicated.

For the splines approach in [3], a sequence of splines is used to generate a path through a given set of waypoints. However, prior knowledge of the waypoints may not be available due to the unknown environment and the kinematic constraints of the robots are not considered in splines. Thus, the trajectory may not be applicable to a specific robot.

In a common cubic spline method, each section of the path could be described by the parametric equations:

$$\begin{aligned} x(u) &= a_x u^3 + b_x u^2 + c_x u + d_x \\ y(u) &= a_y u^3 + b_y u^2 + c_y u + d_y, \end{aligned}$$

where $u \in [0 \quad 1]$. This type of parameterization concentrates on the smooth property at the connection of various segements, rather than the kinematic constraints of the robot. The trajectory obtained by this method may not be feasible for specific types of robots.

In search based methods, A* (proposed in [4]) utilizes a heuristic function to guide the search direction to the goal, thus making it more efficient than the Dijkstra algorithm and ensures an optimal solution from initial point to end point can be found, if one exists. However it requires all of the map information. To deal with dynamic environments, it needs to do a complete recalculation each time the map information is updated, making it inefficient. A typical heuristic index used in A* is:

$$f(n) = h(n) + g(n),$$

where $f(n)$ is the overall cost for a node, $h(n)$ is the cost already spent from the start node to the current node, and $g(n)$ is the estimated cost from the current node to the end node. Generally $g(n)$ can be taken as the Euclidean distance between the current and end nodes.

One improvement of the A* method is found in D* (proposed in [5] and [6]). The D* search algorithm does not require all of the map information. It starts with a priori map and at each time the map data is updated, it invokes a localized A* search to make incremental changes to the path. Its performance is compromised relative to the performance of the A* search. Both search algorithms require heavy computational resources and do not take a kinematic model into account.

By acknowledging the limitations of these techniques, we can improve on these methods by leveraging this information and create an approach that determines a real-time collision-free path for a UAV. In this chapter a parametric solution is proposed to address the limitations of the above techniques. The kinematic constraints are considered, resulting in a class of smooth trajectories. A solution is proposed to design a local decentralized cooperative control for a group of UAV to fly along an arbitrary set of waypoints.

2 Trajectory Planning

The objective of trajectory planning is to find a feasible and smooth trajectory that leads the UAV along its starting waypoint to its final waypoint. In this chapter, trajectory planning is based on the following kinematic model of UAV:

$$\begin{aligned} \dot{r}_x &= v_{r1}\cos(r_\theta) \\ \dot{r}_y &= v_{r1}\sin(r_\theta) \\ \dot{r}_\theta &= v_{r2}, \end{aligned} \tag{1}$$

where (r_x, r_y) are the world coordinates of the UAV, r_θ is the heading angle, v_{r1} is the longitudinal velocity, and v_{r2} is the angular velocity.

2.1 Parameterized Feasible Trajectories

By analyzing the kinematics model described by (1), it can be established that the trajectory is defined by some smooth function $r_y = f(r_x)$. Given initial and final conditions $q_0 = (r_x^0, r_y^0, r_\theta^0)$ and $q_f = (r_x^f, r_y^f, r_\theta^f)$, the model imposes four constraints on the trajectory. That is, the position and first derivative of each end has to match the boundary value. Thus, when the trajectory is parameterized by a polynomial, it should have at least four coefficients. To achieve a class of trajectories, the coefficients could be more than four. In this application, the trajectory is parameterized by a $4th$ order polynomial. That is,

$$r_y = a_0 + a_1 r_x + a_2 r_x^2 + a_3 r_x^3 + a_4 r_x^4, \tag{2}$$

where a_0, a_1, a_2, a_3 are coefficients to be solved and a_4 is free. Given the boundary conditions q_0 and q_f the solution to the coefficients are:

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = (B)^{-1}(Y - Aa_4),$$

where

$$B = \begin{bmatrix} 1 & r_x^0 & (r_x^0)^2 & (r_x^0)^3 \\ 0 & 1 & 2r_x^0 & 3(r_x^0)^2 \\ 1 & r_x^f & (r_x^f)^2 & (r_x^f)^3 \\ 0 & 1 & 2r_x^f & 3(r_x^f)^2 \end{bmatrix}, \quad Y = \begin{bmatrix} r_y^0 \\ \tan(r_\theta^0) \\ r_y^f \\ \tan(r_\theta^f) \end{bmatrix}, \quad A = \begin{bmatrix} (r_x^0)^4 \\ 4(r_x^0)^3 \\ (r_x^f)^4 \\ 4(r_x^f)^3 \end{bmatrix}.$$

2.2 Trajectory Planning for Avoiding Dynamic Obstacles

To deal with the changing environment, as the new obstacle information becomes available, the parameterized trajectory given by (2) may require updates. The updating could be satisfied by a piecewise polynomial parametrization. Let T be the time for a UAV to complete its maneuver from the initial configuration q_0 to its final configuration q_f, and T_s be the sampling period, such that $\bar{k} = T/T_s$ is an integer. When $k = 0$, the initial condition is q_0. For $\bar{k} > k > 0$, the initial condition is given by $q_k = (r_x^k,\ r_y^k,\ r_\theta^k)$, the terminal condition is always q_f. By using the new initial condition, the path planning method described in the previous subsection can be used for real-time replanning as k increases. In the latter part of this chapter, all the notations with superscript k or subscript k indicate they are in the kth sampling period.

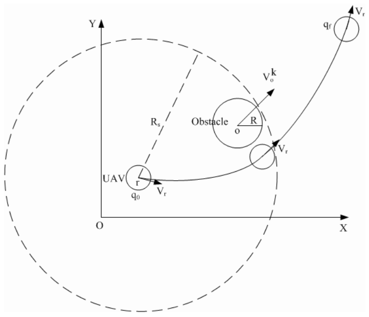


Fig. 1. A UAV in the presence of moving obstacle

Figure 1 illustrates a UAV moving from q_0 to q_f. The radius of the UAV envelope, r, and the sensing range, R_s are known. At the beginning of the kth sampling period, there is a moving obstacle in the sensing range of the UAV, the radius of the obstacle envelop is R with center at $(x_k,\ y_k)$. In this sampling period, its velocity is v_o^k and assumed to be linear, and the values for the data is obtained by onboard sensors. The trajectory (2) is rewritten as:

$$r_y = a_0^k + a_1^k r_x + a_2^k r_x^2 + a_3^k r_x^3 + a_4^k r_x^4. \tag{3}$$

The obstacle avoidance criterion is:

$$(r_y - y_k - v_{o,y}^k \tau)^2 + (r_x - x_k - v_{o,x}^k \tau)^2 \geq (r+R)^2, \tag{4}$$

where $\tau = t - (t_0 + kT_s)$ for $t \in [t_0 + kT_s,\ t_0 + T]$.

According to the results in Sect. 2.1,

$$[a_0^k \ \ a_1^k \ \ a_2^k \ \ a_3^k]^T = (B^k)^{-1}(Y^k - A^k a_4^k), \tag{5}$$

where

$$B^k = \begin{bmatrix} 1 & r_x^k & (r_x^k)^2 & (r_x^k)^3 \\ 0 & 1 & 2r_x^k & 3(r_x^k)^2 \\ 1 & r_x^f & (r_x^f)^2 & (r_x^f)^3 \\ 0 & 1 & 2r_x^f & 3(r_x^f)^2 \end{bmatrix}, \quad Y^k = \begin{bmatrix} r_y^k \\ \tan(r_\theta^k) \\ r_y^f \\ \tan(r_\theta^f) \end{bmatrix}, \quad A^k = \begin{bmatrix} (r_x^k)^4 \\ 4(r_x^k)^3 \\ (r_x^f)^4 \\ 4(r_x^f)^3 \end{bmatrix}.$$

It is not necessary to consider the collision avoidance criterion for all $t \in [t_0 + kT_s,\ t_0 + T]$. Since the collision may only happen when the UAV's x (or y) coordinate is within a certain range. Specifically, the potential collision range obtained from the x coordinate is when $r_x \in [x_k + v_{o,x}^k \tau - r - R,\ \ x_k + v_{o,x}^k \tau + r + R]$. From this condition, a potential collision time interval could be solved as $[\underline{t}^*,\ \bar{t}^*]$. It is only in this time interval that the collision avoidance condition is checked.

Substituting (3) and (5) into (4), one obtains the following inequality:

$$g_2(r_x, k)(a_4^k)^2 + g_1(r_x, k, \tau) a_4^k + g_0(r_x, k, \tau)|_{\tau = t - t_0 - kT_s} \geq 0, \tag{6}$$

for all $t \in [\underline{t}^*,\ \bar{t}^*]$, where

$$\begin{aligned} g_2(r_x, k) &= [r_x^4 - h(r_x)(B^k)^{-1}A^k]^2 \\ g_1(r_x, k, \tau) &= 2[r_x^4 - h(r_x)(B^k)^{-1}A^k][h(r_x)(B^k)^{-1}Y^k - y_k - v_{o,y}^k \tau] \\ g_0(r_x, k, \tau) &= [h(r_x)(B^k)^{-1}Y^k - y_k - v_{o,y}^k \tau]^2 + (r_x - x_k - v_{o,x}^k \tau)^2 - (r+r)^2 \\ h(r_x) &= [1 \ \ r_x \ \ r_x^2 \ \ r_x^3]. \end{aligned}$$

Inequality (6) describes the adjustable coefficient a_4^k, and as long as the chosen a_4^k satisfies this inequality, the obstacle is avoided. For multiple moving obstacles, each obstacle would impose a constraint similar to (6) on a_4^k. When a_4^k satisfies all the constraints simultaneously, all obstacles are avoided.

Figure 2 shows the actual path that the UAVs travelled during the search phase, where the small dots represent static obstacles.

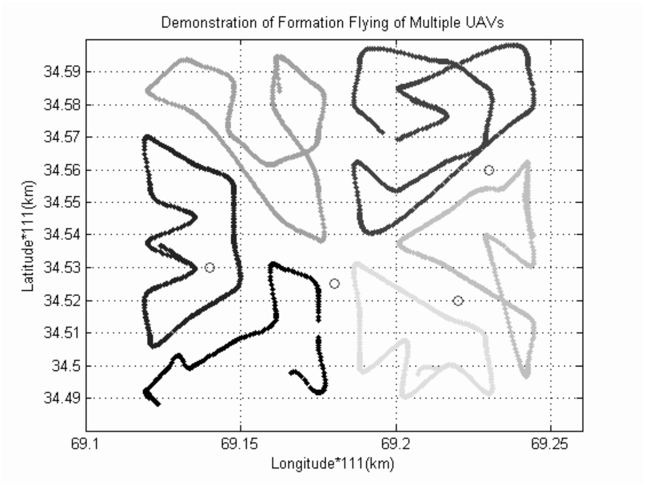


Fig. 2. Trajectory generated in search phase

3 Cooperative Control

In recent years there has been rapid progress in the study of cooperative and formation control for a group of mobile autonomous robots. The reason for this is that cooperatively controlled multiple robots have the potential to complete complicated tasks with a higher efficiency and failure tolerance, such as coordinated navigation to a target, coordinated terrain exploration and search and rescue operations.

Motivated by the flocking behavior of birds in flight, Reynolds introduced a computer animation model for cohesion, separation, and alignment in [10]. Subsequently, a simple discrete-time model (Vicsek model) was given in [11] for the heading alignment of autonomous particles moving in the plane. Simulation results verified the correctness of the Vicsek model. More recently, a theoretical explanation of Vicsek's model was presented in [12] using results from graph theory. The conditions on the connectivity of undirected sensor graphs are given for overall system convergence. This result was extended to networks with directed sensor graphs in [13], [14].

One recent development on designing decentralized local cooperative control is based on matrix theory. Up until now, less restrictive, but successful results have been established in [7]. Given a group of robots that can be feedback linearized into a certain form and their sensing communication matrix satisfies a sequentially complete condition, their production results in a matrix with identical rows, where all the state errors of the group of robots converge, and thus cooperative control is achieved.

3.1 Objectives of Cooperative Control

In general, the control objective for cooperative control is to make the states (or error states) of a group of dynamical systems converge to the same steady state. When applied to formation flying, a group of UAVs converge to a formation when the error states from a group of desired trajectories converge to zero. This is because the states of the group of UAVs in the kinematics model are exactly their position and heading.

3.2 Cooperative Control Algorithm

In order to simplify the design procedure, define the following diffeomorphic state and control transformations

$$\phi_1 = r_x + L\cos(r_\theta), \quad \phi_2 = r_y + L\sin(r_\theta),$$

and

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \cos(r_\theta) & -L\sin(r_\theta) \\ \sin(r_\theta) & L\cos(r_\theta) \end{bmatrix} \begin{bmatrix} v_{r1} \\ v_{r2} \end{bmatrix}.$$

The UAV model can be transformed into the single integrator model as follows with the stable internal dynamics

$$\dot{\phi} = v, \tag{7}$$

where $\phi = [\phi_1, \phi_2]^T$ and $v = [v_1, v_2]^T$.

For an arbitrary path H, a formation can be defined by using its Frenet frame $F_H(t)$, which moves with the path. Let $e_1(t) \in \Re^2$ and $e_2(t) \in \Re^2$ be the orthonormal base of $F_H(t)$, and $\psi^d(t) = [\psi_1^d(t), \psi_2^d(t)] \in \Re^2$ be the origin of $F_H(t)$. A formation consists of q UAVs in $F_H(t)$, denoted by $\{P_1, \cdots, P_q\}$, where

$$P_i = d_{i1}(t)e_1(t) + d_{i2}(t)e_2(t), \quad i = 1, \cdots, q$$

with $d_i(t) = [d_{i1}(t), d_{i2}(t)] \in \Re^2$ being the desired coordinates for the ith robot in $F_H(t)$. It is clear that the rigid formation can be modeled when $d_i(t)$ is constant. The desired position for the ith robot is then

$$\psi_i^d(t) = \psi^d(t) + d_{i1}(t)e_1(t) + d_{i2}(t)e_2(t). \tag{8}$$

To map (7) into the canonical form proposed in [7], define the following decentralized state transformation

$$x_i(t) = \phi_i - \psi_i^d, \quad v = \dot{\psi}_i^d - \phi_i + \psi_i^d + u_i.$$

It follows that

$$\dot{x}_i = A_i x_i + B_i u_i, \quad y_i = C_i x_i,$$

where u_i is the cooperative control for ith robot, and

$$A_i = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad B_i = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad C_i = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

To capture the nature of information flow, define the following sensing/communication matrix:

$$S(t) = \begin{bmatrix} S_1(t) \\ S_2(t) \\ \vdots \\ S_q(t) \end{bmatrix} = \begin{bmatrix} s_{11} & s_{12}(t) & \cdots & s_{1q}(t) \\ s_{21}(t) & s_{22} & \cdots & s_{2q}(t) \\ \vdots & \vdots & \vdots & \vdots \\ s_{q1}(t) & s_{q2}(t) & \cdots & s_{qq} \end{bmatrix},$$

where $s_{ii} \equiv 1$; $s_{ij}(t) = 1$ if the states of the jth UAV is known by the ith UAV at time t; otherwise $s_{ij}(t) = 0$. The general class of cooperative controls are given in the following expression: for $i = 1, \cdots, q$,

$$u_i = \sum_{j=1}^{q} G_{ij}(t)[s_{ij}(t)y_j], \tag{9}$$

where $s_{ij}(t)$ is the entry in the sensing/communication matrix, G_{ij} is a 2×2 block in gain matrix G that reflects the influence of jth UAV's output to the control of ith UAV, it could be designed in the following form:

$$G_{ij}(t) = \frac{s_{ij}(t)}{\sum_{\eta=1}^{q} s_{i\eta}(t)} K_c, \qquad j = 1, \cdots, q, \tag{10}$$

where the design parameter $K_c \in \Re^{2\times 2}$ is a constant, nonnegative, and row stochastic matrix.

3.3 Trajectory Parameterization for Arbitrary Waypoints

In Sect. 3.2, a formation in the Frenet frame $F_H(t)$ is proposed. In most applications, the path of the frame H is not given. Instead, it is desired that the group of UAVs fly through a set of specified waypoints. Suppose a set of waypoints $(w_i, z_i), i = 0, 1, 2, 3$ is given. The following parameterization approach can be used to find the path H.

$$\begin{aligned} z = z_0 &\frac{(w-w_1)(w-w_2)(w-w_3)}{(w_0-w_1)(w_0-w_2)(w_0-w_3)} + z_1 \frac{(w-w_0)(w-w_2)(w-w_3)}{(w_1-w_0)(w_1-w_2)(w_1-w_3)} \\ &+ z_2 \frac{(w-w_0)(w-w_1)(w-w_3)}{(w_2-w_0)(w_2-w_1)(w_2-w_3)} + z_3 \frac{(w-w_0)(w-w_1)(w-w_2)}{(w_3-w_0)(w_3-w_1)(w_3-w_2)}. \end{aligned}$$

Assuming the origin of the frame has a constant overall velocity V (which means the corresponding UAV in the formation has a constant cruise speed), and first waypoint is (w_0, z_0), with a start time at t_0, then the whole timing profile of the Frenet frame $\psi^d(t)$ can be obtained as the following:

$$\psi_1^d(t) = w_0 + \int_{t0}^{t} \frac{V}{\sqrt{1 + (dz/dw)^2}} dt$$

$$\psi_2^d(t) = z_0 + \int_{t0}^{t} \frac{V}{\sqrt{1 + (dw/dz)^2}} dt.$$

Combining with (8), the desired trajectory of all the UAVs can be determined.

4 Simulation

The simulation program was developed in Microsoft Visual Studio .Net 2003 and used with an evaluation copy of the Qt Class library provided by Trolltech. Figure 3 is a flow chart of the simulation program that describes how the code and modules are organized.

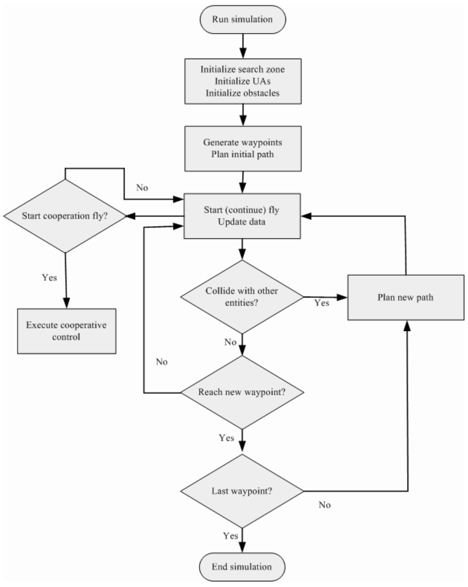


Fig. 3. Simulation platform

4.1 Simulation Prototype

The simulation scenario consists of six UAVs searching within a rectangular area. In the first phase (search phase) the six UAVs perform a complete coverage search over the entire area. In the second phase (cooperative control phase) waypoints are sent to the UAVs via XMPP protocol from the Human Machine Interface (HMI). The six UAVs will converge to a triangular formation and fly along these waypoints.

Table 1. Geographical coordinates of Kabul city

	Longitude(DEG)	Latitude(DEG)
Low-left	69.122650	34.491754
Up-left	69.121734	34.583547
Up-right	69.246323	34.583034
Low-right	69.245865	34.492267

Table 2. Initial configuration of UAVs

	Longitude(DEG)	Latitude(DEG)	Heading(RAD)
UAV 1	69.121734	34.537651	$7\pi/12$
UAV 2	69.163263	34.583547	$-11\pi/12$
UAV 3	69.204793	34.583547	$-11\pi/12$
UAV 4	69.246323	34.537651	$-5\pi/12$
UAV 5	69.204793	34.491754	$\pi/12$
UAV 6	69.163263	34.491754	$\pi/12$

Table 3. Position of static obstacle

	Longitude(DEG)	Latitude(DEG)	Radius(meter)
OBS 1	69.23	34.56	600
OBS 2	69.22	34.52	600
OBS 3	69.14	34.53	600
OBS 4	69.18	34.525	600

Table 4. Waypoints in cooperative fly

	Trajectory 1		Trajectory 2	
WP 1	69.122192	34.5376505	69.1173846	34.5314035
WP 2	69.147018	34.5461272	69.1438258	34.5559345
WP 3	69.196671	34.5648806	69.1938001	34.5779071
WP 4	69.246323	34.5830341	69.2183311	34.5579971

The geographical coordinates of the search area are listed in Table 1. To model static obstacles, moving obstacles with a velocity of zero were used. The initial configuration of the UAV and obstacles are listed in Tables 2 and 3.

In the cooperative flying phase, the two sets of waypoints that the formation should pass are listed in Table 4.

The sensing/communication pattern in the simulation are randomly changing among the following three matrices at each sampling period:

$$S_1 = \begin{bmatrix} 1&0&0&0&0&0 \\ 1&1&0&0&0&0 \\ 0&1&1&0&0&0 \\ 0&0&1&1&0&0 \\ 0&0&0&1&1&0 \\ 0&0&0&0&1&1 \end{bmatrix} \quad S_2 = \begin{bmatrix} 1&0&0&0&0&0 \\ 1&1&0&0&0&0 \\ 1&0&1&0&0&0 \\ 1&0&0&1&0&0 \\ 1&0&0&0&1&0 \\ 1&0&0&0&0&1 \end{bmatrix} \quad S_3 = \begin{bmatrix} 1&0&0&0&0&0 \\ 1&1&0&0&0&0 \\ 0&1&1&0&0&0 \\ 1&0&0&1&0&0 \\ 0&1&0&0&1&0 \\ 0&0&0&1&0&1 \end{bmatrix}$$

The design parameter K_c in (10) is:

$$\begin{bmatrix} 0&1 \\ 1&0 \end{bmatrix}$$

4.2 Simulation Results

To determine the route in the search phase, a minimum number of circles that fit the sensing range of the UAVs are placed in area [9] with the union of these circles covering the entire area. The centers of these circles are the waypoints that will be traveled along to form the route. Next, each UAV determines the waypoints to use. This is done by a Voronoi algorithm, which means each waypoint belongs to the nearest UAV. Finally, each UAV will choose the nearest waypoint as its first waypoint, and then by applying a computational geometry algorithm, the UAV will travel to the nearest unvisited waypoint relative to its current position (going clockwise). This method forces the UAVs to travel in a counterclockwise path. This phase uses the path planing and obstacle avoidance algorithms discussed in Sect. 2.

One case for the search phase and two cases for the cooperative flying phase are simulated. The results for the search phase are shown in Fig. 2. Figures 4 and 5 show the results for the cooperative control phase. By sending different sets of waypoints, the six UAVs fly on different paths in the given triangular formation. After receiving the waypoints, the program first uses the approach discussed in Sect. 3.3 to parameterize the desired trajectories through the set of waypoints, then applies the algorithm presented in Sect. 3. The two figures of the cooperative control phase illustrate that after a transient process, the UAVs gradually converges to their desired trajectory.

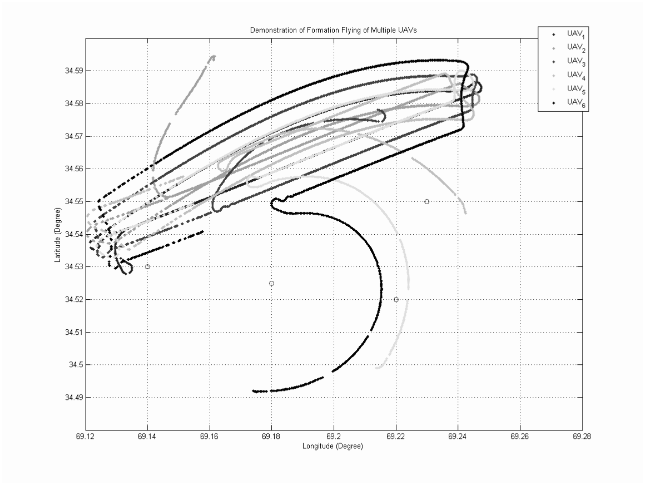


Fig. 4. Trajectory generated in formation fly phase

5 UAS Test Bed

The Unmanned Aerial System Test Bed (UAS Test Bed) is a web-based infrastructure for UAS operational "what if" assessments and development of training strategies developed by L-3 Communications, Link Simulation and Training. It

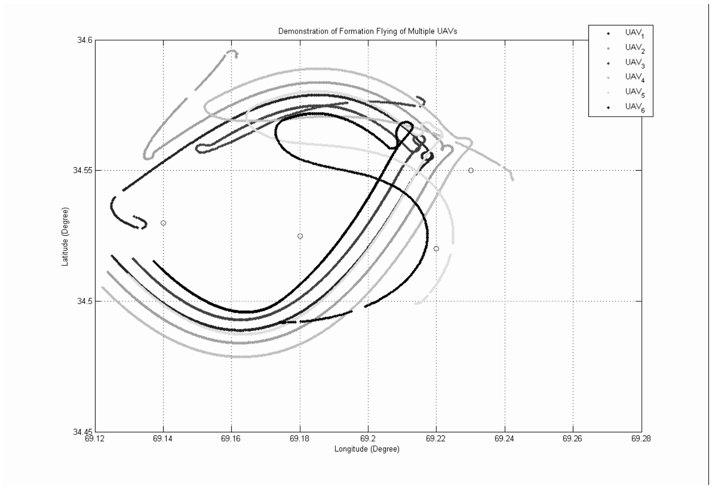


Fig. 5. Trajectory generated in formation fly phase

is built on commercial web, gaming and open source technologies. The Physics-Based Urban Environment provides and manages a common view or state of the virtual environment that the UAS application interacts with. The Multi-Trainer Servers host the UAV, Ground Control Station (GCS) and communication models, which interface to the Physics-Based Urban Environment through a set of APIs. The GCS interfaces to the Human Computer Interface (HCI) through web-based services and protocols. The GCS interfaces to the UAV models through Data Links in the communications layer.

The HCI component provides a tailored trainee interface depending on GCS configuration and desired training position (e.g. vehicle or sensor operator). The HCI's primary displays are used for situation assessment with secondary displays used to assess vehicle or subsystem health. An example generic web based GCS HCI is used to demonstrate the test bed vehicle controls, sensor controls and situation assessment. The HCI is connected to the server side GCS logic, which in turn communicates to UAV models through protocols such as STANAG 4586. Figure 6 illustrates the basic infrastructure in the UAS test bed framework, and Fig. 7 shows the components in the HCI.

The user interacts with this framework and connects to the HCI through a web portal. As a trainee, a user can download and run the HCI while logged into the UAVS web portal, which is their gateway to the simulation. Prior to launching the HCI, the trainee can configure their training session by providing initial parameters and attributes for the vehicle and GCS, and selecting a specific mission. Once the configuration is complete, the trainee can join the simulation as any position they have permissions for. The HCI contains multiple components that are controlled by various roles or positions. A trainee's role is dependent on initial registration parameters and determines group permissions while logged into the web portal. User permissions or authority is based on the training role, and used to determine the available features accessable through the web portal

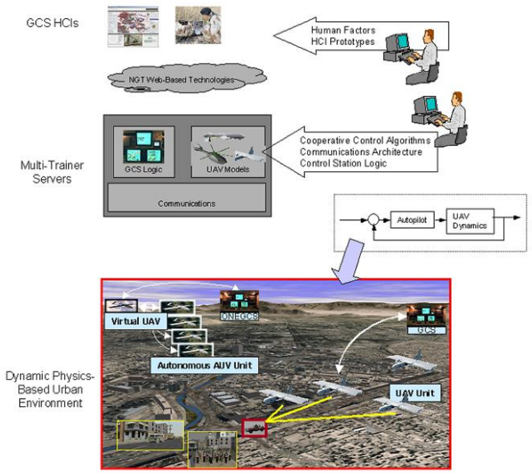


Fig. 6. UAS test bed framework infrastructure

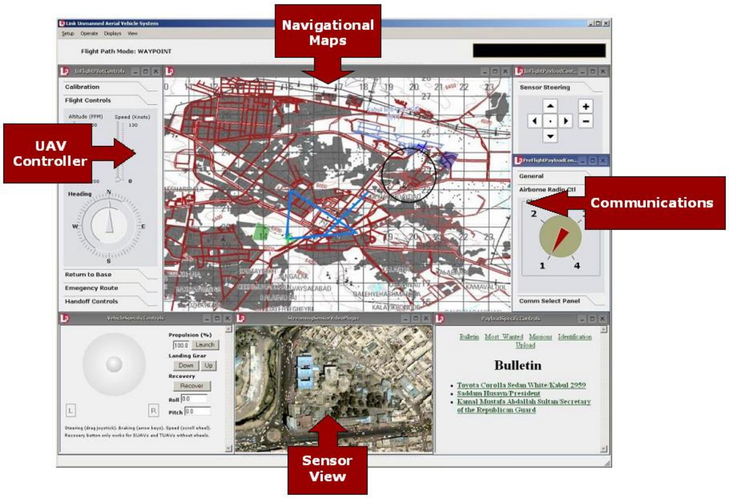


Fig. 7. UAS human control interface

and HCI. Instructors and mission commanders can manipulate all areas of the HCI. Vehicle operators drive the flight controls and execute routes and loiter zones for a single UAV. Sensor operators control the payload steering and zoom for a single UAV. Operators can handoff control to drive other UAVs in the simulation.

Observers can view the UAVs on a map, as well as their sensor output, while planners can only view the map. The map displays all UAVs in the simulation, their inertial states, and all routes and loiter zones that can be executed by the vehicle operators. All users can communicate via voice-over Internet protocol (VOIP) on various channels.

6 UAS Cooperative Control Test Bed Framework

For the UAS Cooperative Control Test Bed, an interface was created, that incorporated the aforementioned cooperative control algorithms, and integrated them into the UAS Test Bed for evaluating their effectiveness using multi-UAV based scenarios. The main areas of integration within the HCI were the moving-map application and the Core UAV Control System (CUCS) communication interface. For the moving-map application, a table of inertial states was added and the ability to interpret additional configuration files. These configuration files provide the locations of static obstacles and targets that are to be displayed on the map. A data translator is used to map the inertial states provided by the cooperative control interface to the UAS interface. Search areas, static obstacles, targets, and pre-defined routes are defined in XML files to be used during the exercise. Figure 8 illustrates the components after integrating the algorithm into the testbed.

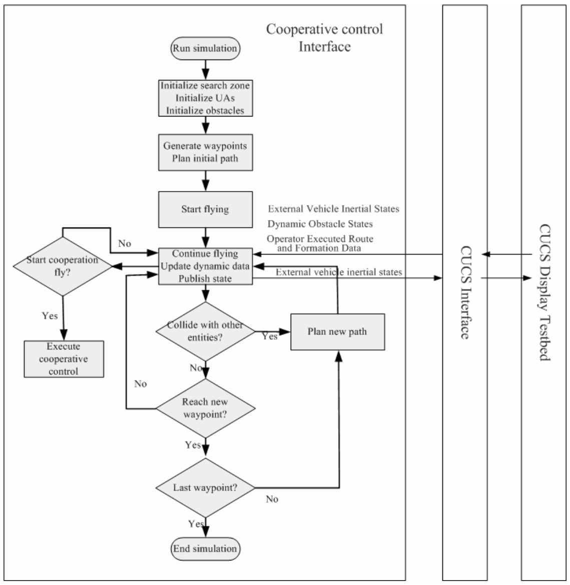
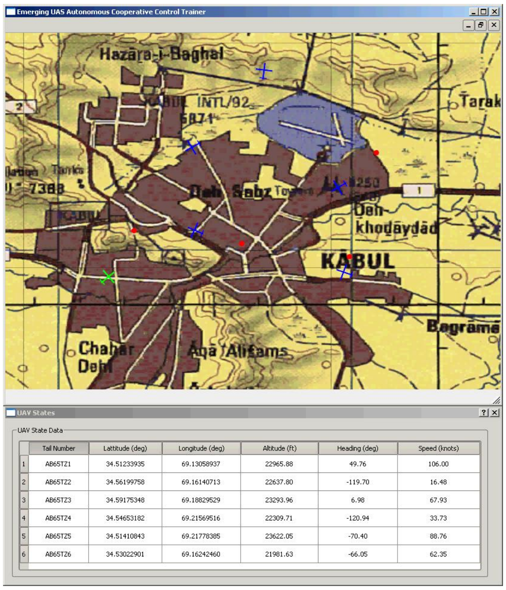


Fig. 8. Integrated simulation platform

Once the program is executed, the UAVs begin searching their respective areas and communicate using the STANAG 4586 [15] standard messages transmitted using the Jabber protocol [16]. The vehicle operator controls the UAVs through a virtual UAV, which can be any one of the UAVs in the formation or the GCS.

This virtual UAV communicates with the formation and they will collectively cooperate to achieve their goal. During the real-time simulation, the following capabilities can be demonstrated:

– Planning the Coverage Search Path
  Once the exercise begins, a search area and the location of static obstacles and targets are uploaded to each UAV. The UAV then generates an initial route that will cover their entire search area, while avoiding the static obstacles. If there are any known dynamic obstacles in the path, the UAV's initial route will reflect this and avoid these entities. Figure 9 shows a snapshot of the working scenario of HCI during the searching phase. The trajectories of the UAVs correspond to Fig. 2.



UAV State Data

	Tail Number	Lattitude (deg)	Longitude (deg)	Altitude (ft)	Heading (deg)	Speed (knots)
1	AB65TZ1	34.51233935	69.13058937	22965.88	49.76	106.00
2	AB65TZ2	34.56199758	69.16140713	22637.80	-119.70	16.48
3	AB65TZ3	34.59175348	69.18829529	23293.96	6.98	67.93
4	AB65TZ4	34.54653182	69.21569516	22309.71	-120.94	33.73
5	AB65TZ5	34.51410843	69.21778385	23622.05	-70.40	88.76
6	AB65TZ6	34.53022901	69.16242460	21981.63	-66.05	62.35

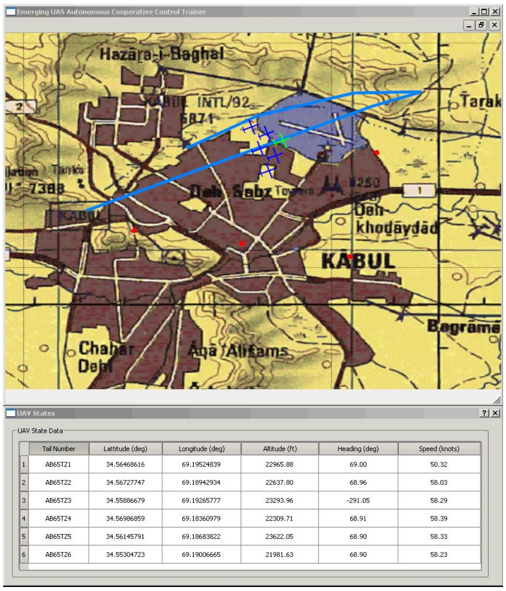
Fig. 9. UAVs flying pre-defined search area

– Real-Time Trajectory Generation
  As the UAVs search they continually communicate with each other their current inertial state and obtain inertial states from other vehicles. Using this information, the UAVs re-plan their route to avoid flying into one another while avoiding obstacles.
– Dynamic and Static Obstacle Avoidance
  During the initial planning and re-planning of the search route, the UAV trajectory will avoid known static obstacles in its path. Other vehicles' inertial states, not limited to UAVs, are also communicated to the HCI's CUCS interface. The UAV uses these locations and velocities to continuously re-plan their route to avoid these dynamic obstacles during search and formation flying.

- Convergence to Specified Targets

  The descriptions of specified targets are uploaded to each UAV upon start of the simulation. The UAV sensor will scan the search area for these targets. Once the target is identified, the UAV will communicate this to the other UAVs. After either all targets have been located or all UAVs have searched the area, the UAVs will converge to the specified targets and begin a loiter pattern.
- Formation Fly Along a Specified Route

  Routes are described in XML and can be uploaded to the virtual UAV and executed. Once the route is executed, the UAVs will generate a new path to fly this route in a formation specified by the XML. The route planning is a cooperative process between all the UAVs in the formation. They communicate their routes to the virtual UAV and re-plan the routes collectively in order to successfully fly the specified route in formation while avoiding obstacles. Figure 10 shows how the group of UAVs flow along a specified set of waypoints, their trajectories correspond to Fig. 4.



	Tail Number	Lattitude (deg)	Longitude (deg)	Altitude (ft)	Heading (deg)	Speed (knots)
1	AB65TZ1	34.56468616	69.19524839	22965.88	69.00	50.32
2	AB65TZ2	34.56727747	69.18942934	22637.80	68.96	58.03
3	AB65TZ3	34.55886679	69.19265777	23293.96	-291.05	58.29
4	AB65TZ4	34.56986859	69.18360979	22309.71	68.91	58.39
5	AB65TZ5	34.56145791	69.18683822	23622.05	68.90	58.33
6	AB65TZ6	34.55304723	69.19006665	21981.63	68.90	58.23

Fig. 10. UAVs flying in formation along a pre-defined route

Given autonomous and cooperative control of multiple UAVs is an emerging capability, there are no real world examples illustrating the operational concepts or training procedures. The integration of cooperative control algorithms with the UAS test bed gives us a platform and ability to perform "what if" scenarios necessary to understand the implications of these capabilities to assess mission performance, operating policy and potential training gaps.

References

1. Jen-Hui Chuang.: Potential-based modeling of three-dimensional workspace for obstacle avoidance. IEEE Trans. on Robotics and Automation, **14** (1998) 778-785
2. K.J.Kyriakopoulos, P.Kakambouras, and N.J.Krikelis.: Potential fields for nonholomic vehicles. Proceedings of the IEEE International Symposium, (1995) 461-465
3. Judd K.B. and Mclain T.W.: Spline based path planning for unmanned air vehicles. AIAA Guidance, Navigation, and Control Conference and Exhibit, volume AIAA-2001-4238, Montreal, Canada, Aug 2001
4. Nilsson, N.J.: Principles of Artificial Intelligence. Tioga Publishing Company, 1980
5. A. Stentz.: Optimal and efficient path planning for partially-known environments. IEEE International Conference on Robotics and Automation, May 1994
6. A. Stentz.: The Focussed D* Algorithm for Real-Time Replanning. Proceedings of the International Joint Conference on Artificial Intelligence, August 1995
7. Z. Qu, J. Wang, and R. A. Hull.: Cooperative control of dynamical systems with application to autonomous vehicles. Submitted to IEEE Transactions on Automatic Control
8. Z. Qu, J. Wang, and C.E.Plaisted.: A New Analytical Solution to Mobile Robot Trajectory Generation in the Presence of Moving Obstacles. IEEE Transactions on Robotics, **20** (2004) 978-993
9. Yi Guo, Z Qu.: Coverage control for a mobile robot patrolling a dynamic and uncertain environment. 5th World Congress on Intelligent Control and Automation, Hangzhou, China, Jan. 2004
10. Reynolds,C.W.: Flocks, herds, and schools: a distributed behavioral model. Computer Graphics (ACM SIGGRAPH Conference Proceedings), **21** (1987) 25-34
11. Vicsek, T., Czirok, A., Jacob, E.B., Cohen, I. and Shochet, O.: Novel type of phase transition in a system of self-driven particles. Physical Review Letters, **75** (1995) 1226-1229
12. Jadbabaie, A., Lin, J., and Morse, A.S.: Coordingation of groups of mobile autonomous agents using nearest neighbor rules. IEEE Trans. on Automatic Control, **48** (2003) 988-1001
13. Lin, Z., Brouchke, M., and Francis, B.: Local control strategies for groups of mobile autonomous agents. IEEE Trans. on Automatic Control, **49** (2004) 622-629
14. Moreau, L.: Leaderless coordination via bidirectional and unidirectional time-dependent communication. Proceedings of the 42nd IEEE Conference on Decision and Control, Maui, Hawaii 2003
15. North Atlantic Treaty Organization: Standard Interfaces of the UAV Control System (UCS) for NATO UAV Interoperability. April, 2004
16. Dodson Catherine: Jabber Technical White Paper. Jabber.com, Inc, April, 2000