

Drone Formation Control System Real-Time Path Planning

Manuel Soto¹

UNITECH, White Sands Missile Range, NM, 88002

Patricia A. Nava²

The University of Texas, El Paso, TX, 79968

Luis E. Alvarado³

The Proteus Corporation, White Sands Missile Range, NM, 88002

This study focuses on the implementation and demonstration of the Real Time Path Planner (RTPP). It is an AI guidance system that was developed for an operational DoD unmanned aerial target control system. The RTPP is tested using the 6-DOF target simulator of Drone Formation Control System (DFCS). The RTPP uses the A* algorithm to generate the obstacle free routes. The data used for the A* graph, which represents the terrain map, was obtained from the National Elevation Dataset. UAV flight trajectories developed by this system undergo testing in a DoD flight simulator. An MQM-107D simulated drone is used for the testing. The partition size of the terrain map is varied within the study to obtain an optimal partition size. These tests analyze the flight trajectories produced by the RTPP, and the effect of varying the A* and the UAV parameters. This study demonstrates that the RTPP produces safe and maneuverable routes for the MQM-107D. Results are confirmed by time plots of key flight parameters recorded during the 6-DOF UAV simulation runs. The time plots show that the target did not have any difficulty flying the computer-generated pattern. It also shows that the target flew over benign terrain as predicted by the RTPP to reach its destination. This study demonstrates the innovative benefits and functionality added by Intelligent Systems theory to the real-time path planning and navigation task via the DFCS system. The RTPP works as designed, producing safe and flyable flight patterns for the drone.

Nomenclature

3-D	=	3 dimensional
A*	=	Route finding algorithm, pronunciation: A-star
AGL	=	Above Ground Level in ft
AGL _m	=	minimum Above Ground Level in ft
AI	=	Artificial Intelligence
BAH	=	Barometric Altitude Hold
d_i	=	drone index
DFCS	=	Drone Formation Control System
DME	=	Distance Measuring Equipment
DoD	=	Department of Defense
DOF	=	Degrees of Freedom
Drone	=	UAV
DTED	=	Digital Terrain Elevation Data
ECEF	=	Earth Center Earth Fixed (geocentric) coordinate system
ENU	=	East North and Up coordinate system
f_i	=	flight pattern index
$f(n)$	=	evaluation function
FP	=	Flight Pattern
$g(n)$	=	cost function
G	=	32.2 ft/s ²
geocentric	=	ECEF coordinate system

¹ Computer Engineer, UNITECH, Bldg. 335, P.O. Box 339, WSMR, NM, 88002.

² Professor and Chair, Electrical & Computer Engineering, UTEP, El Paso, TX, 79968.

³ Control Engineer, Proteus Corp., Bldg. 335, P.O. Box 339, WSMR, NM, 88002.

geodetic	=	latitude and longitude with respect to equator and prime meridian, respectively, coordinate system
$h(n)$	=	heuristic function
h	=	MSL height in ft
h_a	=	route flight altitude in ft MSL
h_e	=	terrain elevation in ft MSL
IS	=	Interrogator Stations
m	=	terrain map resolution – the number of points in the terrain map
MSL	=	Mean Sea Level in ft
MQM-107D	=	subscale RPV (UAV)
n	=	node
n_i	=	current node
n_j	=	designates either the previous node or the next node
NED	=	National Elevation Dataset
N_z	=	Target Normal Acceleration in Gs
q	=	quantization value – the horizontal and vertical distance between points on the terrain map
V_z	=	Target Vertical Acceleration in ft/s
r_t	=	turn radius
RF	=	Radio Frequency
RFM	=	Radio Frequency Module
RPV	=	Remotely Piloted Vehicle
RTPP	=	Real Time Path Planner
s_i	=	flight pattern segment index
SHOT	=	Speed Hold on Throttle
Target	=	UAV
Topographic	=	Earth surface projection to a flat plane
UHF	=	Ultra High Frequency
v_g	=	ground speed in ft/s
WGS 84	=	World Geodetic Systems 1984
WSMR	=	White Sands Missile Range
x_f	=	final route x location
x_i	=	initial route x location
x_{min}	=	minimum x value on terrain map in ft
x_{max}	=	maximum x value on terrain map in ft
y_f	=	final route y location
y_i	=	initial route y location
y_{min}	=	minimum y value on terrain map in ft
y_{max}	=	maximum y value on terrain map in ft
z	=	distance between the $z=0$ plane and a reference object in ft
ψ	=	true heading in degrees
ψ_f	=	final true heading
ψ_i	=	initial true heading
ψ_m	=	magnetic heading
ϕ	=	bank angle in degrees

I. Introduction

An Unmanned Aerial Vehicle (UAV) guidance system was created for Department of Defense (DoD) target control systems. It uses theories from the field of Artificial Intelligence (AI) to solve existing problems being faced by guidance system operators. The system aims to solve problems caused by inflexibilities in four areas of UAV guidance: flight trajectory geometry, obstacle avoidance, real-time operation, and operator work load reduction. One problem addressed is that of specialized knowledge being required for creating flight trajectories. Another problem addressed is the collision risk that exists when flying at low elevations over mountainous terrain. Also addressed is the need for flight trajectories to be developed before the flight mission due to the dangers that exist when changing their geometry in real-time. The last problem addressed is that of minimizing labor intensive tasks.

The guidance system described in this paper has the following characteristics: it takes in four types of inputs, it contains four internal functional modules, and it returns one type of output. The following are the four types of inputs: digital terrain elevation data, UAV flight parameters, no-fly zones, and trajectory start and termination coordinates. When a safe path is obtained, the output is a safe flight trajectory. When no safe path is found, no guidance is provided by the system. The four internal functional modules perform the following jobs: path finding, obstacle avoidance, path selection, and path smoothing. The inputs are used by the guidance system's path-finding and obstacle-avoidance modules to produce a path; if a path is found, it is sent to the path selection module for ensuring that a safe flight trajectory can be obtained from it. If the path found is safe, it is sent to the path-smoothing module and processed to create a safe flight trajectory for UAV guidance.

II. Drone Formation Control System

A. DFCS Overview

The DFCS is a mature control system located in Bldg. 335 at White Sands Missile Range (WSMR) New Mexico. Its mission is to control single and/or multiple unmanned full-scale and sub-scale targets for the Army. These targets are used by the Army to test and evaluate new combat systems. The DFCS can automatically control the QF-4 full-scale target, the BQM-34A, MQM-107D and MQM-107E sub-scales, and a wide variety of ground targets such as the M60 and T-72 tanks and the M-812 five-ton truck. The DFCS system consists of 6 drone control consoles, 2 master control consoles, two central RISC type computer control subsystems (CCS), onboard data link transponders and control subsystems, two co-located interrogator subsystems (ISC), 10 DME¹¹ ground interrogator stations (IS) and 4 UHF RFM units located at strategic mountain peak locations to cover most of White Sands Missile Range airspace.

B. Navigation

DME Target navigation is accomplished by RF data link using distance-measuring techniques. Slant propagation time from the IS's to each drone is measured and used to calculate slant range. The Computer Control Subsystem interrogates each drone ten times per second to calculate the space position of the drone. The DFCS Navigation weighted-least-squares filter also mixes the IS slant range information with other down linked target altitude information such as radar and barometric altimeter data and target inertial information to optimize the tracking accuracy of the system. DFCS also uses information such as detailed terrain map data to augment target altitude estimation. GPS Target navigation is accomplished by processing the GPS TSPI data down linked by the UHF subscale targets. The DFCS GPS navigation filter provides inertial navigational data during data link outages and very accurate estimation of target velocity and acceleration.

C. Guidance

DFCS can automatically guide any target or a formation of targets over a pre-defined flight pattern, which can be manually modified in real time. The guidance philosophy is to direct each formation (up to 6 aerial targets per formation and up to 48 ground targets per formation) to a moving space point referred to as the "rabbit". The rabbit is generated in real time based on the desired flight path. DFCS operators also have the capability to vary in real time, mission control parameters such as: target and formation velocity, altitude and cross track deviations from the predefined flight pattern, formation spacing between the targets and the minimum airspeed and altitude allowed. DFCS also has the unique capability to control and synchronize up to 6 different drone formations. Throughout history, DFCS has achieved full-scale QF-4 target formation presentations with a minimum separation of 150 feet between targets flying 150 feet above ground level at Mach 0.9.

D. Control Capabilities

DFCS has the capability to control up to 6 full-scale targets such as the QF-4 during a single mission. The airborne subsystem in each target is interrogated ten times per second by the ground computer through the data link Interrogator Subsystem (IS). Each interrogation contains approximately 10 proportional channels and 80 discrete channels on the uplink and 30 proportional channels and 80 discrete channels on the downlink telemetry. DFCS has two basic control modes of operation; manual mode and automatic mode. In manual mode a drone controller sitting at one of the drone control consoles manually flies the target by monitoring the displayed target telemetry data and issuing desired drone attitude and speed control commands. The onboard computer receives and processes these up-linked commands and generates the corresponding servomechanism commands *remote fly by wire*. In automatic mode, the ground computer replaces the human in the loop. It transforms drone flight pattern guidance corrections into drone attitude commands and engine throttle command settings. DFCS also has the capability to automatically

takeoff and land the QF-4 target at Holloman Air Force Base, located in Alamogordo, New Mexico. The DFCS control system also has numerous pre-programmed emergency maneuvers that the target will automatically execute if the ground computer *thinks* the full-scale target is in trouble. These maneuvers are automatic terrain avoidance, collision avoidance, minimum airspeed and Data Link Loss maneuvers. The DFCS also has the capability to fly full scale and subscale targets at very low altitudes over rough

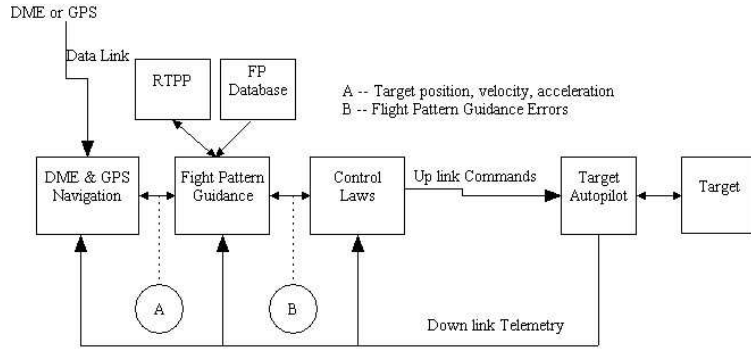


Figure 1. DFCS block Diagram.

terrain by using the recently implemented Derived Radar Altitude Penetration Enhancement (DRAPE) algorithm. This algorithm uses high fidelity terrain database and calibrated barometric altimeter data or GPS to derive a radar altitude of the target in real time. A nap-of-the-earth algorithm is then used to control the altitude of the target at the desired altitude. DFCS also has the capability to calibrate barometric altitude data using radar telemetry or GPS down-linked information.

Figure 1 shows that DFCS receives target position data from DME equipment or from GPS. These data are processed by the DFCS navigation filters. The navigation data is then sent to the flight pattern guidance algorithm. This guidance algorithm uses flight pattern information either from the flight pattern data base or from the Real Time Path Planner (RTPP). The guidance errors are then computed. That is the cross track, along track, and altitude errors between the rabbit and the target. The control laws use these errors to generate the target up-link commands. The target transponder receives the ground station up-links and sends them to the target autopilot. The target autopilot closes high frequency control loops such as pitch and pitch rate control, roll attitude and roll rate control and others. The target autopilot then sends telemetry data to the ground station and DFCS uses these data to close some of the ground control loops.

III. Simulation Setup

The *rational agent*⁸ description will be used to define the criteria for testing the AI UAV guidance system that was created for this study. First, assign the UAV as the *agent*, and the Real Time Path Planner (RTPP) as the *agent program*. Let the terrain be the *environment*; and let the *task environment* be succinctly stated as follows: to guide the UAV safely according to the user entered specifications. These user entered specifications are the following items. The first set of specifications are the route's initial and final location. The second specification is the UAV MSL flight altitude; it must remain constant for the entire flight. The third specification is that the agent program must select a route that maintains a minimum AGL between the UAV and the terrain beneath it, for the entire flight. The fourth set of specifications involve the specified true headings of the start and goal locations. Specifically, the path must use the specified true headings assigned for exiting the start location, as well as the true headings for entering the goal location. The fifth and final specification: the agent program must build a flight trajectory that the agent can use for the entirety of the flight; this flight trajectory will comply with UAV turn radius constraints for the flight's maximum ground speed. The *performance measure* for this task environment is discussed next.

Regarding the performance measure; Russel and Norvig⁸ define it as follows: "The *performance measure* evaluates the behavior of an *agent* in an *environment*." The RTTP expected behavior was defined above, but the *environment* and the *performance measure* have not been defined. The performance measure for this study is to be done with MATLAB analysis. The measurements will be an assessment of how closely the UAV followed the user specifications on the assigned flight trajectory in real-time. An analysis will also be done on the flight trajectories recommended by the RTTP. The UAV flight data will be obtained from a DoD flight simulator; simulations will be of an MQM-107D UAV following the RTTP recommended flight trajectory. The *environment* for this study has two components: real-time flight trajectory requests, and the WSMR terrain. The real-time flight trajectory requests will be provided to the RTTP by the DFCS MQM-107D flight simulator through a UDP computer network connection. The WSMR terrain will be provided, through data files, by a computer program called the Terrain Map Creator (TMC). This program creates quantized topographic maps that partition the terrain into evenly-spaced squares; each square is assigned the highest MSL elevation value within its area.

A. Terrain Environment

The East North and Up (ENU) coordinate system is used at DFCS. The coordinates can be represented as (x, y, z, h) . z is defined with respect to an imaginary flat plane that is tangent to the earth surface at the WSMR center, which is the origin of the DFCS ENU coordinate system. h is the MSL elevation (or altitude); it takes into account the curvature of the earth, which makes it very useful for measuring UAV AGL. The $(x, y, z, h) = (0 \text{ ft}, 0 \text{ ft}, 0 \text{ ft}, 0 \text{ ft})$ origin in the ENU coordinate system is at 33.0834 degrees North and 106.3339 degrees West. This point is 72.4112 ft MSL above the WGS 84 reference ellipsoid. All altitudes and elevations are passed as h to the RTPP, and all coordinates represented as three-tuples are the previously defined four-tuple, with the z coordinate eliminated for simplicity. For example, an object at $(20,000 \text{ ft}, 20,000 \text{ ft}, 500 \text{ ft})$, is not at 500 ft above the surface of an imaginary flat plane, it is at 500 ft above the surface of the earth.

The TMC generates the terrain environment for the RTPP. It partitions the map into discrete (x, y, h) locations. This is a two steps process. First, the terrain is quantized by selecting the map's (x, y) coordinates. Second, the real-world elevation value, h , is assigned to each (x, y) coordinate. The h is value is obtained from a digital terrain elevation database. The partitioning of the map is done as recommended by Mitchell⁵. He states: "The simplest surface to model is a plane, and the simplest nontrivial terrain map on a plane is a binary partitioning of the plane into regions that are obstacles and regions that are traversable by the vehicle." The map squares resulting from the partitioning are the size of the maximum UAV turn radius that will be used during the flight. The determination of whether a map square is an obstacle or traversable terrain is done by the RTPP. However, the RTPP uses h as one of its criteria for making this decision.

B. Real Time Environment

DFCS has a very complex and realistic 6 DOF flight simulator. The DFCS simulation program includes modeling of the vehicle aerodynamics and inertia data including the interaction between the physical environment and the guidance and control system. Also modeled are the atmosphere, the data link subsystem, the onboard autopilot, vehicle sensors, engine forces and moments, and servo subsystems. The flight simulator has the following aircraft models that can be used for simulations: QF-4, BQM-34A, MQM-107D, and MQM-107E. These models can simulate all the flight phases of the DFCS mission: takeoff, formation flight, maneuvering, recovery and landing. For this study, the MQM-107D was used, and only the takeoff phase was considered in the analysis. The take-off phase established a real-time environment by having the UAV airborne when requesting guidance from the RTPP.

IV. Guidance Issues: Relationship between A* Route and Terrain Map Resolution

The RTPP provides guidance by processing waypoints into flight patterns. The flight pattern development process restricts the terrain maps to have a map resolution quantization value equal to, or greater than, the UAV turn radius. The A* algorithm uses these terrain maps to produce a route that is composed of (x, y, h) coordinates joined by line segments. This path is processed into waypoints; and the waypoints are used to create the flight patterns. This section provides an explanation of why and how this process is carried out.

A. Processing of A* Paths

Not all A* routes can be used for UAV guidance because some routes generated, although theoretically sound, are difficult, or impossible, to process into maneuverable UAV flight trajectories. A* routes can be viewed as functions dependent on the following variables: map geometry (i.e. partition shape and size), heuristic function, and cost function. The map geometry affects the size of the individual line segments that compose the A* route. Consider the case of a route generated from an (x, y, h) map that is partitioned into squares, that could have consecutive line segments that oscillate between two true headings due to obstacles. For the oscillation between diagonal headings, the path will also oscillate, forming a zig-zag pattern.

The real thing to consider is how the UAV would meet waypoints produced by using the map described above. For example, A* could return a route, headed east, whose line segments oscillate between true headings of 45 degrees and 135 degrees. Even if the UAV is traveling at its minimum airspeed, it would have problems meeting all waypoints because meeting one (or two in succession) will cause it to overshoot many of the following waypoints. As a quantitative example, consider a UAV traveling on the east bound oscillating route described above. Assume that this UAV has a ground speed, v_g , of 650 ft/s, with corresponding minimum turn radius of about 7815 ft. It can meet the first two waypoints easily if it enters the path at the true heading made by the first points. However, because of the turn radius resulting from $v_g = 650 \text{ ft/s}$, the next 80 waypoints will be overshoot.

Important questions concerning the use of a map with *appropriate resolution* that is *correct* for the geometry of the map partitions must be addressed. One such question is the following: Will increasing the *map partition size*,

which is a *square* for this study, fix the overshoot problems and thus allow the A* paths to be maneuverable by a UAV? The answer to this question depends on the UAV MSL and the variance in terrain elevation, and on this problem's requirement that the UAV maintain constant MSL during the flight. There are trade-offs between map partition size and UAV MSL. For instance, if guiding a UAV over terrains whose elevations vary, such as terrains that contain mountains, increasing the partition size will restrict to the UAV to flying at higher MSL altitudes. This situation arises from the method used to create the terrain maps: in order to avoid collisions between the terrain and the UAV, each map partition is assigned the highest elevation MSL within the partitions area.

One goal of this study is to obtain guidance that produces flight trajectories that go in between mountains. Thus the solution posed by increasing partition size is not acceptable. Also, configuring the map geometry to lessen waypoint overshoot is not enough to remove the risk and uncertainty caused by waypoints when dealing with UAV guidance over obstacle-rich terrain. It seems that the problem is not only in the partition size, but also in using waypoint navigation. The unappealing property of waypoint navigation is that the waypoints are used as guides only; where overshooting and undershooting the waypoints to compensate for meeting the following waypoints is common. A better solution to the problem would do two things: (1) reduce UAV position ambiguities (e.g. overshoot and undershoot); and (2) determine the minimum map partition size. Such a solution would allow the UAV to use lower MSL flight altitudes with reduced risk while providing a flight trajectory that will allow the UAV to stay on the recommended flight trajectory by complying with UAV dynamics.

B. Guidance: Waypoints or Flight Patterns

Waypoints were considered as the output of the RTPP; however, *flight patterns* seemed a better solution for two reasons. One, flight patterns are better at minimizing the terrain collision risk, and two, they allow the UAV to fly at lower MSL altitudes safely over obstacle rich terrain. Three important differences between waypoints and flight patterns were considered for distinguishing which guidance method was better suited for this application. First, different from waypoint navigation, which is the use of a set of ordered coordinates as guides, flight patterns do not use way points. Flight patterns are composed of arc and line segments that take into account the UAV turn radius. Second, flight patterns are continuous as opposed to waypoints, which are discrete. This feature is useful in determining if high terrain is in the path of the UAV. And third, a waypoint, which can be seen as a point in 3-D space, is used as a flight goal by the UAV, one at a time; whereas a flight pattern, which can be conceptualized as a track in 3-D space, is a trajectory that the UAV is assigned to follow.

Also, flight pattern error measurements seem better suited to this application when compared to waypoint error measurements. Error in flight patterns is not measured by overshoot, undershoot, or if the UAV was at a particular location with respect to one point. The error is measured for the entire flight pattern, from the point in time the UAV was assigned to be on the flight pattern, until it was completed. As with waypoints, overshoot and undershoot measures are not applicable to flight patterns. What is applicable, that is of interest to this study, are the altitude and cross track error. Altitude error is how much distance the UAV was above or below the flight pattern; cross track error is the perpendicular displacement between the UAV and the flight pattern.

C. Terrain Map Creator

The TMC produces *terrain maps* of similar geometry; however, the properties of each map can vary. These properties are the following: location, shape, size, resolution. The map geometry is obtained by partitioning the map area into *squares*; squares of *equal length and width*, rectangular map partitions are not used in this study. Maps for different geographical areas within the boundaries of WSMR can be produced. The NED image's borders extend beyond the WSMR limits, hence geographical maps, within the NED image boundaries can also be produced (e.g. if WSMR extensions are to be used). The *size* and *shapes* of the geographical areas are determined by the *map limits*, which are: x_{\min} , y_{\min} ; x_{\max} , and y_{\max} . These limits are used to produce a quantized topographic map for the RTPP. The *map resolution*, m , is the number of (x, y, h) coordinates in the terrain map for a geographical area outlined by the limits, x_{\min} , y_{\min} ; x_{\max} , and y_{\max} , of the terrain map. It is determined by the *quantization value*, q , of the map; the quantization value, q , is the horizontal (and vertical) distance between each point on the map; the diagonal distance between points is $\sqrt{2}q$. For a fixed geographical area, the *map resolution* is inversely proportional to the *quantization value* (i.e. as the quantization value, q , increases; the map resolution, m , decreases when the geographical area is fixed). Terrain maps are quantized to (x, y, h) coordinates that are separated by the same distance, q , vertically and horizontally. q has an *optimal value*. It is the UAV turn radius, r_t , computed at the maximum ground speed, v_g , that will be used for the flight. The method that was used to discover that r_t is optimal for obtaining terrain maps is explained next.

D. Very Low Terrain Map Resolutions

Selecting a map resolution forces a trade-off. The trade-off is between attaining less collision risks while giving up flying at the lowest MSL altitude, h_a ; as map resolution, m , decreases the lowest MSL flight altitude, h_a , increases. The reason for this situation is in the method used for creating the terrain maps (i.e. assigning the highest peak within the square's area). The partitioning of an area taken from WSMR can be used to explain why this situation occurs. Since, the problems caused by very large map resolutions were explained when waypoint overshoot was discussed, the following examples focus exclusively on the trade-off of giving up low MSL flight altitudes when using terrain maps with very low resolutions. In the examples, an area of 158,400 ft by 475,200 ft (30 miles by 90 miles) is extracted from WSMR. The area is partitioned into very large squares for simplicity, and, assumed elevation, h_e , values will be used to illustrate the trade-off (NOTE: WSMR elevations vary from approximately 3,000 ft MSL to approximately 10,000 ft MSL.).

For the first example, create a map for the RTPP that uses quantization value, $q = 52,800$ ft (10 miles), for the map's partition size. The TMC creates a topographic map that contains $m = 27$ squares. It also assigns the maximum elevation, h_e , within the square's boundaries in order to eliminate unexpected collisions. Now, to illustrate the trade-off between safety and minimum MSL altitude, h_a , assume that each square has as its highest elevation, a mountain peak at 9000 ft MSL. This will result in the TMC assigning an elevation of $h_e = 9000$ ft MSL to each square. Hence, the minimum MSL flight altitude before setting the minimum AGL has to be above 9000 ft MSL. Then, if a minimum AGL of 500 ft is assigned, the flight MSL altitude will be at $h_a = 9500$ ft MSL. This example shows a very safe flight, where overshoot is not a problem; but it came at the cost of flying at a very high MSL altitude.

For the second example, a terrain map of half the quantization value of the above example, which is $q = 26,400$ ft, will be used. The TMC partitions the area of 158,400 ft by 475,200 ft into 108 squares. Now, assume that 27 peaks have an elevation of 9000 ft. MSL and that the rest of the terrain has a maximum elevation of 7000 ft MSL, and assume that the 9000 ft MSL peaks occur in different squares. The TMC will produce a map that has 27 squares with elevations of 9000 ft MSL, and 101 squares with elevations of 7000 ft. MSL. If a minimum AGL of $AGL_m = 500$ ft is used, the RTPP will search for UAV flight trajectories in squares that have assigned elevations as low as $h_e = 7500$ ft. MSL.

E. Optimal Terrain Map Resolution

The purpose of varying the terrain map partition size was to obtain an optimal quantization value, q , and an optimal terrain map resolution value, m , that would make A^* paths useful for real-time UAV guidance. Two problems resulting from not having an optimum map resolution value have been discussed in detail. One was that selecting a map resolution that is lower than necessary increased the minimum MSL flight altitudes, h_a . The other problem was that selecting small quantization values returned waypoints that are separated at values less than the space needed for the UAV to perform turns, which causes the UAV to overshoot the waypoints. From the observed problems, it seems that the minimum distance between the centers of the map partitions should be the maximum UAV turn radius, $r_{t,max}$, that will be used during the guidance period.

Creating a map that has the quantization value, q , equal to the UAV turn radius, r_t , requires that the UAV ground speed, v_g , be known for the entire time frame for which guidance will be provided. The reason for this is that the turn radius, r_t , and the UAV ground speed, v_g , are directly proportional to each other as shown in Equation 1.

$$r_t = \frac{v_g^2}{a_z \tan(\varphi)} \quad (1)$$

where

- r_t is the turn radius, in ft; and
- a_z is the expected target vertical acceleration at 1G; and
- 1G = 32 ft/s²; and
- φ is the desired bank angle, in degrees; and
- v_g is the ground speed; in ft/s.

Note: the vertical acceleration of 1 G may not be true if target experiences air turbulences during flight.

A map that has the turn radius, r_t , value as quantization value, q , when computed with equation 1, will have points that are horizontally and vertically separated by $r_{t,max}$. The diagonal distance between the centers of the partitions will be greater; this separation distance will be $\sqrt{2} r_{t,max}$.

Selecting the turn radius as the optimum quantization value allows the UAV to access to lower MSL flight altitudes, h_a , but the overshoot problem is not completely removed. However, one major benefit arises. It is if the ground speed, v_g , is reduced, the UAV has access to even lower MSL flight altitudes while using a flight trajectory that it can stay on. The use of the ground speed being determined to create the map partition size allows the UAV to navigate the sharp turns within the mountains. However, the overshoot problem still occurs when the returned route contains three diagonal links in succession, whose size are exactly the maximum turn radius, $r_{t,max}$; the resulting error is a sharp turn (a discontinuity), which occurs at the third waypoint. To keep the access to the lower valued MSL flight altitudes, h_a , this problem will be fixed with the A* cost function, $g(n)$. Routes that contain discontinuities will be discarded iteratively, and when a valid route is obtained, the iteration process will terminate, and the waypoints will be used to generate a flight pattern. Thus, it is maintained that the q is optimal at $r_{t,max}$.

V. Design

This section describes the technologies and theories that were used to design and implement the RTPP, and the TMC program from which the terrain portion of the RTPP environment is extracted. The TMC process for creating elevation maps of geographical locations for real-world UAV guidance will be discussed. The performance measure of how effectively the RTPP guided the UAV is done by using MATLAB analysis, and will be described in the simulations and results section of this document.

A. Design Overview

Two programs were created for this study – the TMC and the RTPP. They function independent of each other; however the files created with the TMC are used by the RTPP. Using one program to create terrain maps and the other to provide UAV guidance has several benefits. First, it becomes easier to create custom maps. This feature is useful for creating terrain maps of different geographical locations (e.g. from other continents) or for different terrain map resolutions of the same geographical area with different quantization value, q , which is necessary since the RTPP uses the $q = r_t$ (UAV turn radius) to create flight patterns that can be implemented by the UAV. The RTPP flight trajectory creation module requires that the map's quantization value, q , be inversely proportional to the UAV ground speed, v_g . The RTPP uses q to calculate the maximum turn radius, $r_{t,max}$, that will be required for navigation over obstacle rich areas, where collision risk is high. The higher resolution maps (i.e. lower quantization value for the same geographical area) are used for lower ground speeds, where lower v_g is required for navigating sharp turns.

B. RTPP Route Detection and Obstacle Avoidance Algorithm Selection

A study was conducted for selecting a route-finding solution. From the study, the fields most likely to yield a practical solution were neural networks¹, dynamic programming², and state-space search³. Further research narrowed the field to state-space search, namely blind *versus* heuristic search. The blind-search methods, such as depth-first-iterative-deepening or bi-directional searches³, were discarded because of their tendency to exhaust computer resources, such as memory, before finding the solution. From heuristic search, A* was selected for the following reasons: it guarantees to find the shortest path if a solution exists⁴, it has been used with Defense Agency data⁵, and it has been used for UAV path finding^{6,7}. In AI literature, A* is described as being *complete*; this means that if a path exists, A* will find it (on condition that computer resources are not exhausted).

Also, there are several variations of the A* algorithm that have mitigated some known problems. The following A* variations limit computer memory usage: MA* (memory-bounded A*), SMA* (simplified MA*), and IDA* (iterative-deepening A*). These algorithms are recommended for solving problems resulting from insufficient computer memory while finding paths in *large-scale problems*⁸. However, they do have their drawback, such under using the memory available to them, and taking longer to obtain solutions⁸. From the state-space methods described, memory and time issues provide the highest risk. From the available algorithms, A* seemed the most promising, and if computer resource exhaustion problems arose, A* would be modified to one of its variations.

C. Terrain Map Elevations

The TMC elevation values, h_e , are obtained from the National Elevation Dataset (NED)⁹. This elevation dataset has a resolution accuracy of 30 meters, and it has the property of providing the highest peak within those 30 meters, and the elevation error is less than one meter. A NED elevation-query function was available at the beginning of this study. Its inputs were latitude, and longitude using geodetic coordinates. The DFCS ENU coordinate system, which is a flat plane, was converted to geodetic coordinate system, which is a spheroid. The geodetic conversions translated the DFCS ENU plane at $z = 0$ to an area of the geodetic spheroid centered at -106.3339 degrees longitude and 33.0834 degrees latitude.

The following are the intermediary geographic conversions performed by the TMC to translate the DFCS ENU coordinate system to the geodetic coordinate system: topocentric-to-geocentric and geocentric-to-geodetic. These transformations use the World Geodetic System 1984 (WGS 84)¹⁰ ellipsoid model. The World Geodetic Systems created this model using satellite measurements of the earth. The first geographic conversion performed by the TMC is topocentric-to-geocentric; this conversion takes the *topocentric* DFCS ENU z plane at elevation $z = 0$ to the three dimensional *geocentric* coordinate system, whose origin is (0 ft, 0 ft, 0 ft) at earth's center. The resulting coordinates are three-tuples on a sphere that uses earth's center as its origin. The second geographic conversion is geocentric-to-geodetic. The geodetic coordinates uses spherical coordinates to represent locations on the earth. This system uses the equator as the vertical reference point and the prime meridian as the horizontal reference point.

D. Real Time Path Planner Inputs

How and when flight patterns and waypoints are produced by the RTPP depends on the input mode and on the input values: the input modes are off-line and real-time, and input values are the UAV flight profile properties. Real-time mode is used when providing guidance to DFCS or when it is turned on during diagnostic testing. An example of diagnostic testing would be in using the RTPP to evaluate its own performance when using an experimental heuristic function. The default mode of operation is off-line; but real-time mode can be made the default by setting a configuration file. The UAV flight profile inputs can be entered through a GUI, through the diagnostic system, or through a UDP computer network link. The flight profile properties, which are the following listed values, are passed by the RTPP to the A* algorithm:

- UAV MSL flight altitude h_a , for the entire route
- UAV minimum AGL, AGL_m , for the entire route
- UAV maximum ground speed, $v_{g,max}$, for the duration of the desired route
- Start location, (x_i, y_i) , of the route
- Start true heading, ψ_i , of the route
- Destination location, (x_f, y_f) , of the UAV route
- Destination true heading, ψ_f , of the UAV route

Real-time mode is used to load flight profile properties as follows. Real-time mode activates a 10 ms clock that checks a RTPP buffer for the flight profile properties, which are called flight pattern requests. The flight profile properties are the specifications that A* uses for computing the route. Some flight profile properties have default values. Default values do not exist for the start location, (x_i, y_i) , and heading, ψ_i , the destination location, (x_f, y_f) , and heading, ψ_f , and the UAV ground speed, $v_{g,max}$. The RTPP uses the ground speed, $v_{g,max}$, to compute the maximum turn radius allowed and to load the associated terrain map. Default values are assigned to the minimum AGL, which is 165 ft, and for the UAV flight altitude, which is 10,000 ft MSL. The UAV ground speed to UAV turn radius relationship is described in the optimal map resolution discussion. AGL and true headings are used as constraints by A*, which can be explained using set theory. Consider the minimum AGL case. Let the universal set be the set of all available squares from the terrain map, and let the subset of squares whose difference between their assigned elevation value and the UAV MSL flight altitude be equal to or greater than the user specified minimum AGL. A* uses only the subset of squares that meet the minimum AGL to enforce that the route obtained meets specifications. The AGL and the true heading are defined in the following equations:

$$AGL = h_a - h_e \quad (2)$$

where

- h_a is the MSL flight altitude, in feet; and
- h_e is the MSL terrain elevation, in feet.

$$\psi_m = \psi - offset \quad (3)$$

where

- ψ_m is the magnetic heading, in degrees; and
- ψ is the true heading, in degrees; and.
- $offset = 11.25$ degrees in WSMR.

AGL_m and UAV ground speed, $v_{g,max}$ are required for requesting a path, but start and destination headings are not; they are not assigned default values, either. If headings are not provided, the most convenient heading will be chosen by the A* algorithm. However, if headings are specified, A* uses them to search for a path that has the specified start and/or destination true headings. When a true heading is specified, the RTPP snaps the value to one of eight possible values: 0 degrees, 45 degrees, 90 degrees, 135 degrees, 180 degrees, 225 degrees, 270 degrees, and 315 degrees. The ranges are centered at these values; each covers a 45 degree arc from where it is centered. A* uses this quantized true heading to select the next map square (in the case of the start location, or the penultimate map square (in the case of the destination location)).

E. A* Evaluation Function

The A* evaluation function was used to obtain a continuous obstacle-free route by setting the heuristic and cost functions. The heuristic function directs the agent program to the destination location from the start location. The cost function controls the following items: the route's start and destination true headings, obstacle avoidance, it enforces the minimum AGL constraint, and penalizes locations that interfere with flight pattern smoothing. The evaluation function, $f(n_i)$, is the sum of the cost function, $g(n_i)$ and the heuristic function, $h(n_i)$ at a particular node, n_i ; the node represents a location on the terrain map. $h(n_i)$ and $g(n_i)$ are all functions of a node. The heuristic function used for this study is the two dimensional Euclidean distance; it was used because it is a known *admissible* heuristic function. *Admissible* heuristic functions do not over estimate the goal. The following equations identify the evaluation function, heuristic and function, cost function respectively:

$$f(n_i) = h(n_i) + g(n_i) \quad (4)$$

where

- f is the evaluation function; and
- h is the heuristic function; and
- g is the cost function; and
- n_i is the current node; and
- i is the node index.

$$h(n_i) = \sqrt{(x(n_i) - x_f)^2 + (y(n_i) - y_f)^2} \quad (5)$$

where

- $x(n_i)$ is the ENU DFCS x coordinate corresponding to the i^{th} node; and
- $y(n_i)$ is the ENU DFCS y coordinate corresponding to the i^{th} node; and
- x_f is the ENU DFCS x destination coordinate; and
- y_f is the ENU DFCS y destination coordinate; and
- n_i is the current node; and
- i is the node index.

$$g(n_i) = \sum_{i=1}^{n_i} g(n_i) \quad (6)$$

where

- $g(n_i)$ is the cost to get from the start node to the i^{th} node; and
- $g(n_j)$ is the cost to get from the current node to the next node; and
- n_i is the current node; and
- n_j is the next node.

The cost function used for this study returns one of two values:

$$g(n_i) \in \{1, \infty\} \quad (7)$$

If the value returned is 1, the square traversable. If the value returned is ∞ , the square is not traversable.

The evaluation function, $f(n_i)$, uses the heuristic function, $h(n_i)$ and the cost function, $g(n_i)$, to perform the following two tasks: (1) advance in the *correct* general heading of the destination location; and (2) determine if the next location is safe. The heuristic function orders nodes according to distance between the current location (assigned to the node, n_i) and the destination location, (x_f, y_f) . Since we are using the two-dimensional Euclidean distance on the $z = 0$ plane, an implicit assumption is made: that no obstacles exist in-between the current location and the destination location. The cost function, $g(n)$, takes the agent from the current position to the next position. It evaluates if the next location is safe, one square at a time. It does this when the agent is at one of the connecting squares. Squares that have a cost of $g(n_j) = 1$ are safe, and squares that have a cost $g(n_j) = \infty$ are dangerous. Squares that do not have a cost of 1 have the following properties:

- for the route's start location: the true headings between the current location and the next location is not the one specified
- for the destination location: the true headings between the previous location and the current location is not the one specified
- the distance between the terrain and the UAV exceed the minimum AGL specified
- are obstacles (i.e. imminent collisions)
- failed to provide enough space for the UAV turn radius in the previous path.

Nodes that have any of the properties listed above have a cost, $g(n_j)$, of infinity.

VI. Simulation Results

A. Background

The RTPP was linked to DFCS using the UDP protocol via TCP-IP.

The keyboard commands available at the DFCS consoles subsystem to request guidance from the RTPP are listed and described in Table 1. The RTPP uses the anticipated UAV ground speed to determine the turn radius that will be used to generate the flight patterns. The RTPP uses the commanded UAV MSL flight altitude and the minimum AGL parameter to determine a safe flight trajectory between the start and destination locations.

Table 1.		
Keyboard Command	Command Arguments	Description
ENA ASTAR		Enables UDP protocol network communication between the RTPP and DFCS
INH ASTAR		Inhibits UDP protocol network communication between the RTPP and DFCS
PATH ASTAR	d_i	Requests a flight pattern from the RTPP with the specified attributes from the DFCS console subsystem. This command can be used off-line or in real-time. The attributes are as follows: 1.) the drone index, 2.) the initial route position, (x, y) in DFCS ENU coordinates, 3.) the initial true heading for the route, 4.) the flight pattern altitude, h, in ft MSL, 5.) the final route position, (x, y) in DFCS ENU coordinates, 6.) the flight pattern final heading, 7.) the minimum flight pattern AGL distance, and 8.) the expected target ground speed.
	x_i	
	y_i	
	θ_i	
	h_a	
	x_f	
	y_f	
	ψ_i	
	AGL _m	
	v_g	
	d_i	
ADD ASTAR	d_i	Requests a flight pattern from the RTPP that obtains the current UAV location as the initial location, and obtains the final location from the user entered attributes. This command is entered from the DFCS console subsystem. The attributes are as follows: 1.) the drone index, 2.) the final route position, (x, y), in DFCS ENU coordinates, 3.) the flight pattern final heading, 4.) the minimum flight pattern AGL distance, and 5.) the expected target ground speed. This is a real time command; the drone can be airborne when the command is entered. The start position, flight pattern MSL altitude and initial route heading are equal to the position, altitude, and heading of the target at the time the command is entered.
	x_f	
	y_f	
	ψ_i	
	AGL _m	
	v_g	
SEG ASTAR	d_i	Request a flight pattern from the RTPP that obtains the current UAV location obtains the final location from the user entered attributes; the final location is at a the start of an existing flight pattern. This command is entered from the DFCS console subsystem. The attributes are as follows: 1.) the drone index, 2.) the segment index, 3.) the minimum flight pattern AGL distance allowed along the pattern, 4.) the UAV ground speed This is a real time command; the drone can be airborne when the command is entered. The start position, flight pattern MSL altitude and initial route heading are equal to the position, altitude, and heading of the target at the time the command is entered. The final flight pattern heading is the heading of the start segment on the connecting flight pattern.
	s_i	
	AGL _m	
	v_g	

Table 1. DFCS Keyboard Commands. Table 1 lists the keyboard command and described their associated parameters, that were used to obtained flight patterns from the RTPP during this study's MQM-107D simulations.

B. Simulation Runs

This section describes three 6-DOF simulations of the MQM-107D drone using RTPP guidance. The simulations were conducted to evaluate the performance of the RTPP. For each simulation a flight patterns was developed prior to launch with the RTPP. Each of these pre-launch RTPP flight patterns are related by the following criteria: they use the same start and destination location with the following different MSL flight altitudes: 6500 ft MSL, 6750 ft MSL, and 7000 feet MSL. The same start and destination location with different flight altitudes are used to illustrate the A* obstacle avoidance capabilities. To show the real-time capabilities of the RTPP guidance, the first two simulations used the RTPP to obtain flight patterns that connect the UAV in real-time from its current location to start location of the pre-launch generated flight patterns.

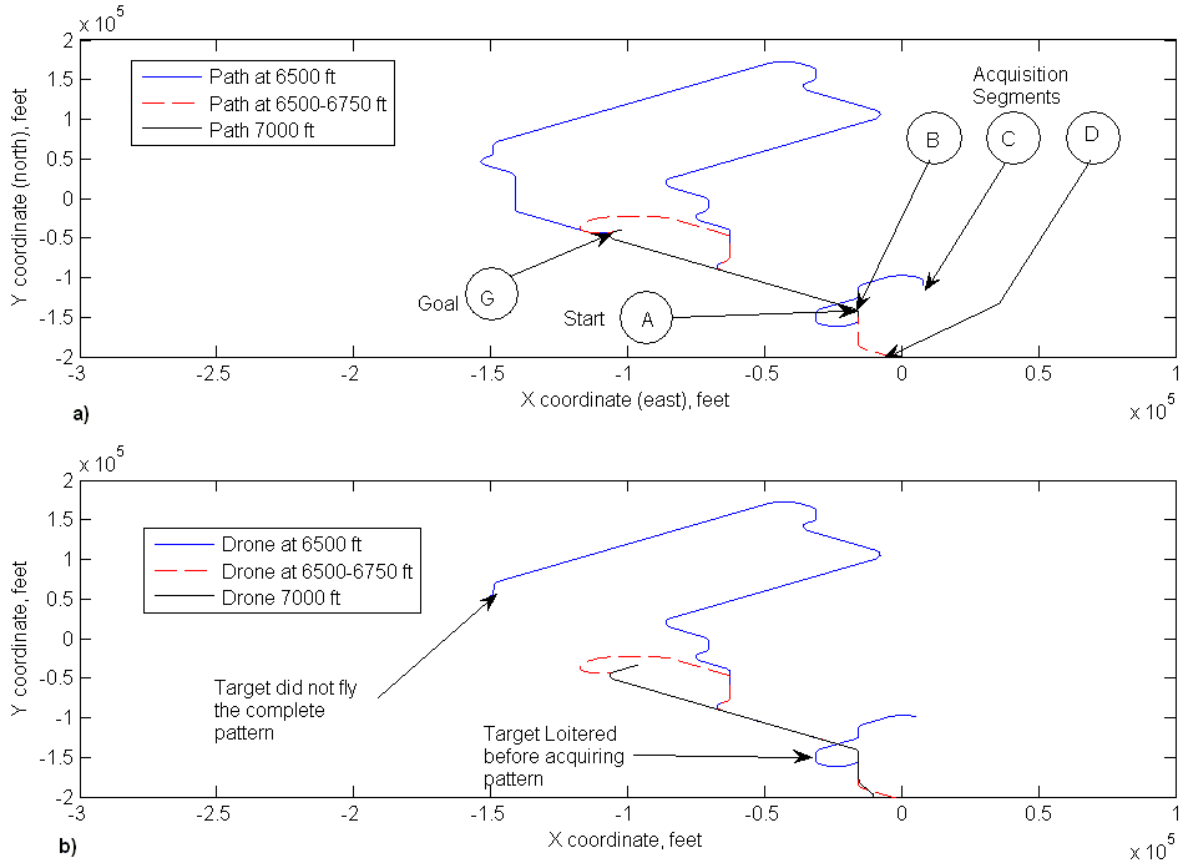


Figure 2. MQM-107D simulation flight trajectories. Figure 2a. Flight paths generated by RTPP, Fig. 2 b. trajectories flown by simulated target.

Figure 2a shows a 2D plot in the DFCS ENU coordinate system of the patterns generated at 6500 feet MSL altitude (blue pattern), 6750 feet MSL (red pattern) and at 7000 feet MSL (black pattern). Figure 2b shows the trajectories followed by the simulated MQM-107D target when they flew the flight patterns.

Labels A and G in Fig. 2a, shows the commanded start and goal positions used to generate these patterns. It is interesting to notice that the 6500 ft MSL pattern (blue) had to go all the way around the mountain to reach its destination point where the 7000 feet pattern (black) goes straight to its destination point. The following subsections describe the target simulations conducted using these three RTPP generated flight patterns:

Figure 3 shows a 3-D plot of the three RTPP generated flight patterns on the WSMR terrain using the DFCS ENU coordinate system. Three flight patterns are shown: the blue, red, and black flight patterns were generated for 6500 ft MSL, 6750 ft MSL, and 7000 ft MSL flight altitudes respectively.

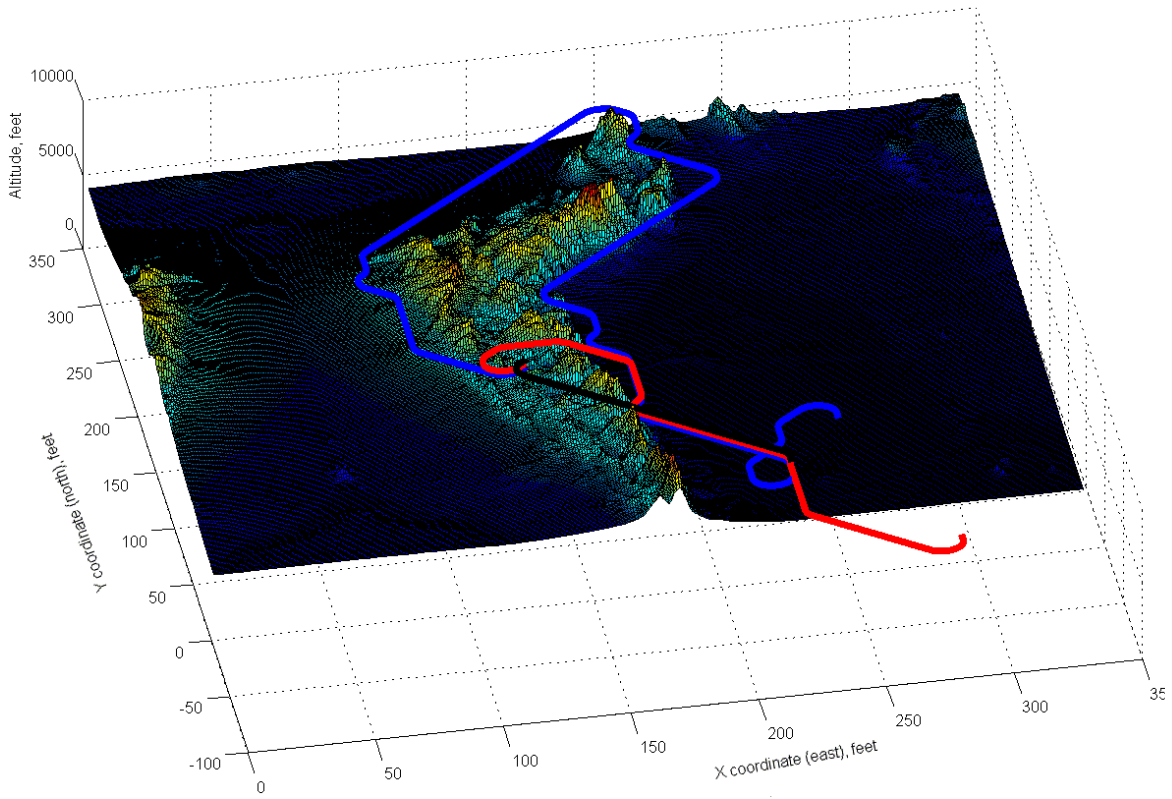


Figure 3. RTPP generated flight patterns over the WSMR terrain. 3D plots of flight patterns generated by RTPP

C. MQM-107D simulation on the 6500 feet MSL blue pattern

The MQM-107D target was manually launched and flown toward the “blue” pattern at approximately 6500 ft MSL at a ground speed of 600 ft/s. The Barometric Altitude Hold (BAH) and the Speed Hold on Throttle (SHOT) control modes were used to manually fly the target to point “C” depicted in Fig. 2. At point “C” the DFCS operator entered the SEG ASTAR keyboard command specifying the goal segment, a 200 feet minimum FP AGL and the expected target speed of 600 feet per second. The RTPP immediately generated a safe and flyable pattern needed by the target to acquire the existing 6500 feet MSL “blue” pattern. Notice that RTPP was sufficiently intelligent to add a loiter to the acquisition pattern to be able to intercept the 6500 feet MSL blue pattern at the correct heading.

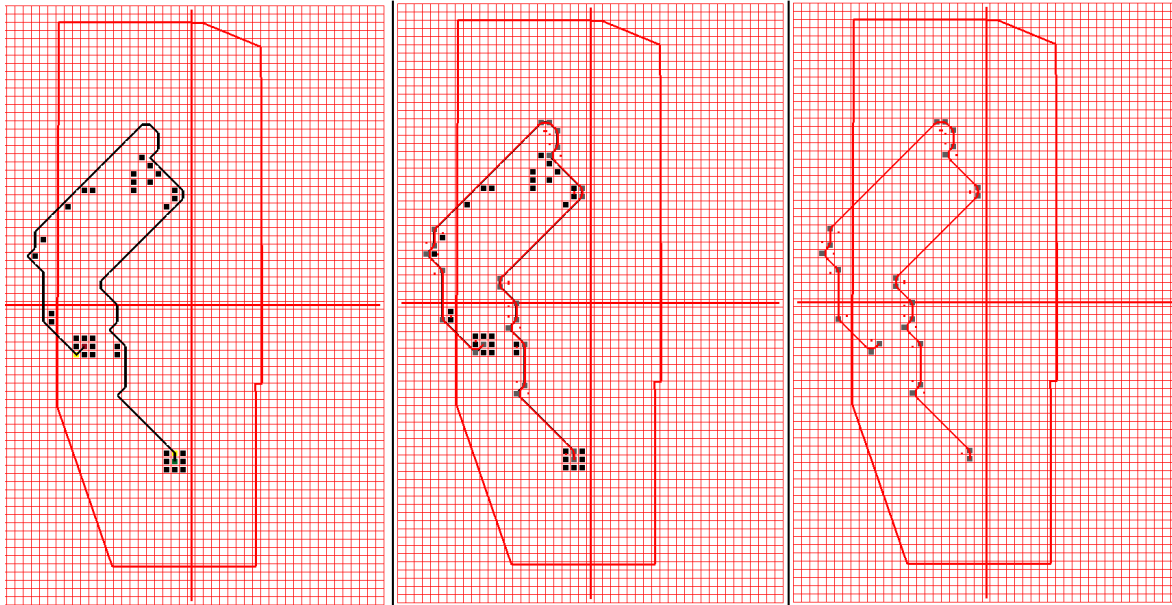
The MQM-107D drone was put in automatic control mode and the target started following the acquired flight pattern. Figure 2, subplot 3, blue trace, shows the trajectory followed by the simulated MQM-107D target. Notice that the simulation of the target was stopped before the target finished flying the whole flight pattern.

Figure 5, subplot 1, shows the terrain MSL altitude beneath the “blue” pattern. Figure 5a shows the target altitude AGL estimated by DFCS using DTED level 2 data and the target MSL altitude. The red trace in Fig. 5b shows the minimum FP AGL altitude allowed. Notice that the target altitude was never below this altitude of 200 feet AGL.

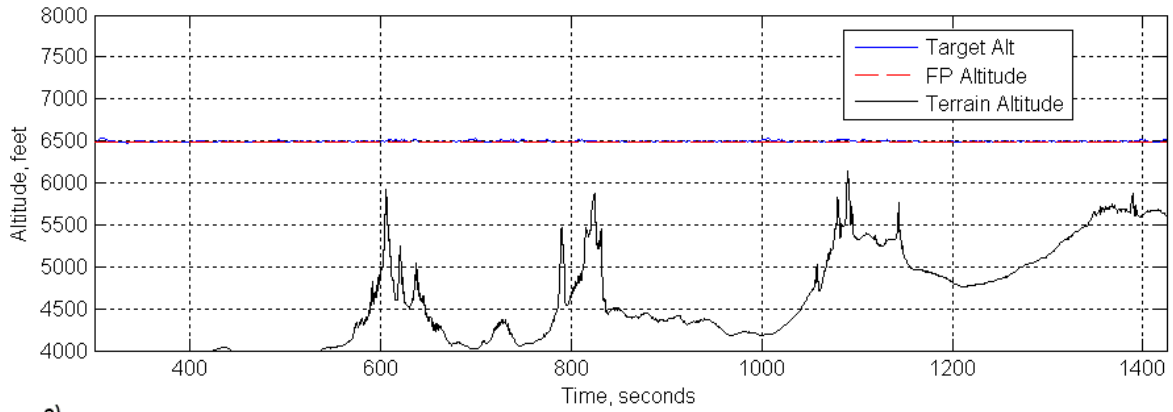
Figure 6 shows a plot of terrain elevation close to the 6500 feet MSL blue pattern. The plot shows the maximum terrain elevations detected inside radii of 1000 feet (black trace), 2000 feet (magenta trace) and 3000 feet (red trace) from points along the generated flight pattern. The blue trace shows the terrain elevation underneath the flight pattern. Figure 6 also shows that, at one point in time, at about 3000 feet from the flight pattern, there was terrain higher than 6500 feet MSL. Therefore, as long as DFCS can maintain the cross track error within 1000 feet, the target will be safe and should be able to fly the generated blue pattern without any problems.

Figure 7a shows a time plot of the target cross track error (i.e. perpendicular distance between the target and the flight pattern) when the target was on the blue pattern. Notice cross track errors of approximately 350 feet during target turns. The reason for these cross track deviations is that the target roll command was limited to 60 degrees for this run and the maximum expected speed was mistakenly set precisely equal to the rabbit speed of 600 feet per second. The target roll angle is plotted in Fig. 7b. Cross track control would be considerably improved by making the commanded (rabbit) speed at least 50 ft/s lower than the expected flight pattern speed. This would insure that the flight pattern turn radii are large enough for the target to stay on the pattern during turns.

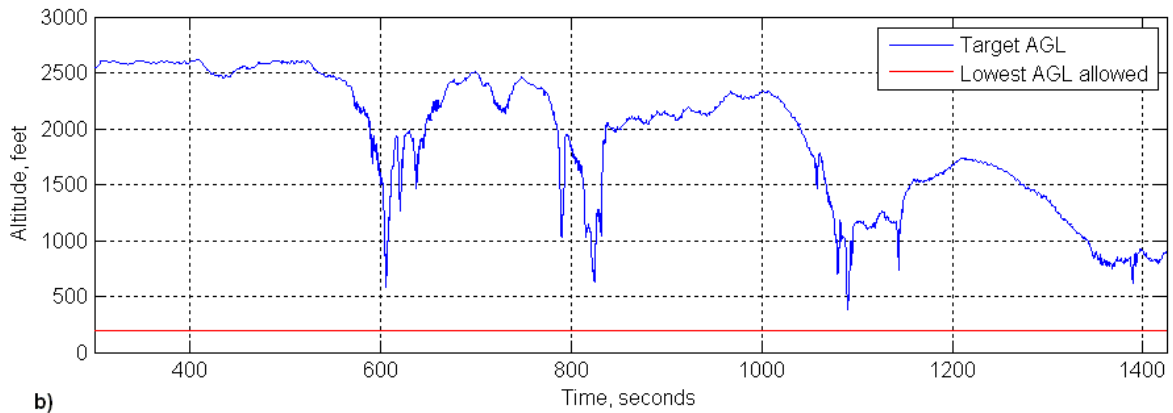
Figure 8a shows along track error. That is the along track distance between the target and the rabbit. Figure 8b shows a time plot of the target ground speed in feet per second (ft/s). Notice the target speed increased during turns because it could not stay on the pattern and had to fly slightly outside the turn segment. Since the rabbit always stays on the pattern, the target had to increase its speed to be able to keep up with the rabbit. Since the expected speed was set exactly equal to the control speed of 600 ft/s, any speed deviation will cause the target to fly outside the pattern during turns. Figure 8c shows a time plot of the target altitude error; that is, the altitude difference between the target and the flight pattern altitude. This plot shows that altitude control was tight and stable during the simulation run.



a) **Figure 4. RTPP Screen shots.** *Figure 4a.A* path, Fig. 4 b.A* path, waypoints and flight pattern, Fig.4c.waypoints and flight pattern*



a)



b)

Figure 5. Figure 5 a. Terrain beneath 6500 feet MSL flight pattern, Fig. 5 b. Target AGL

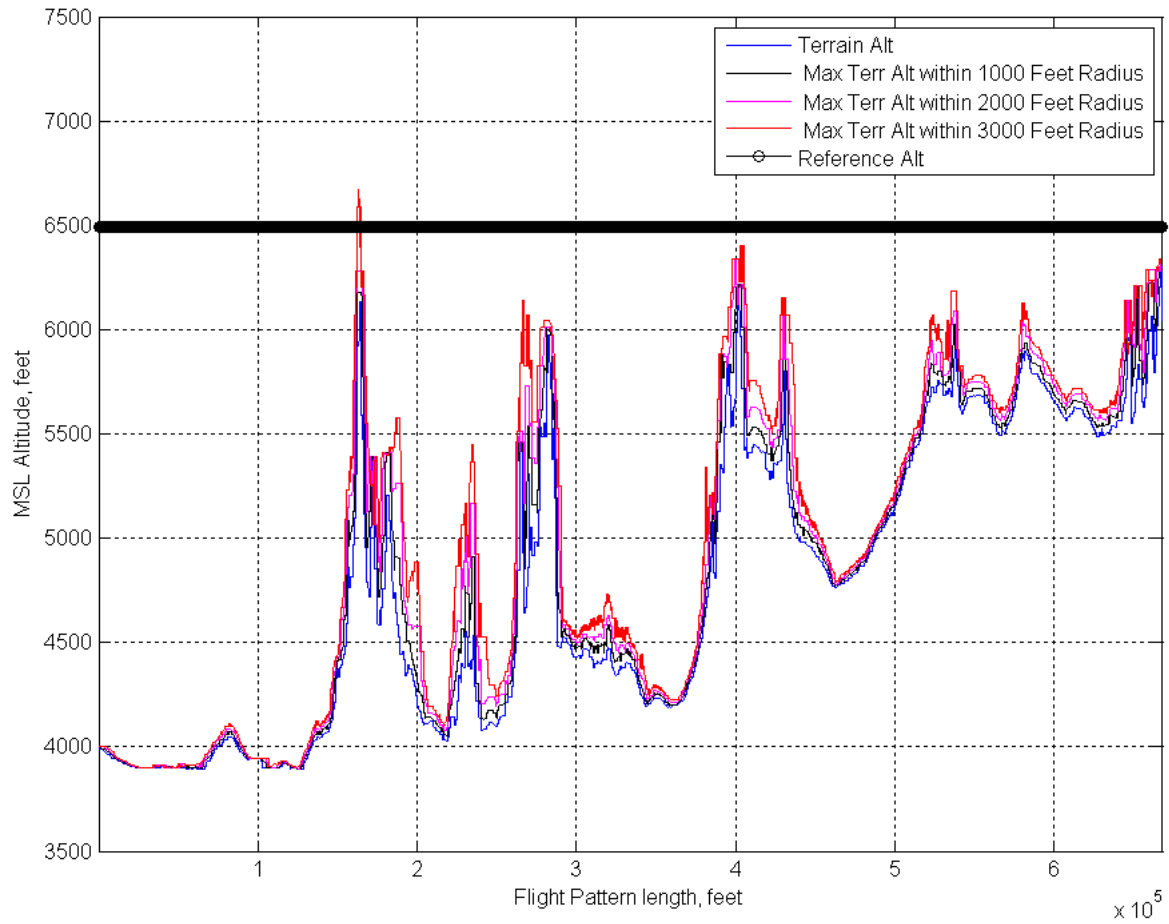


Figure 6. Terrain nearby 6500 feet FP.

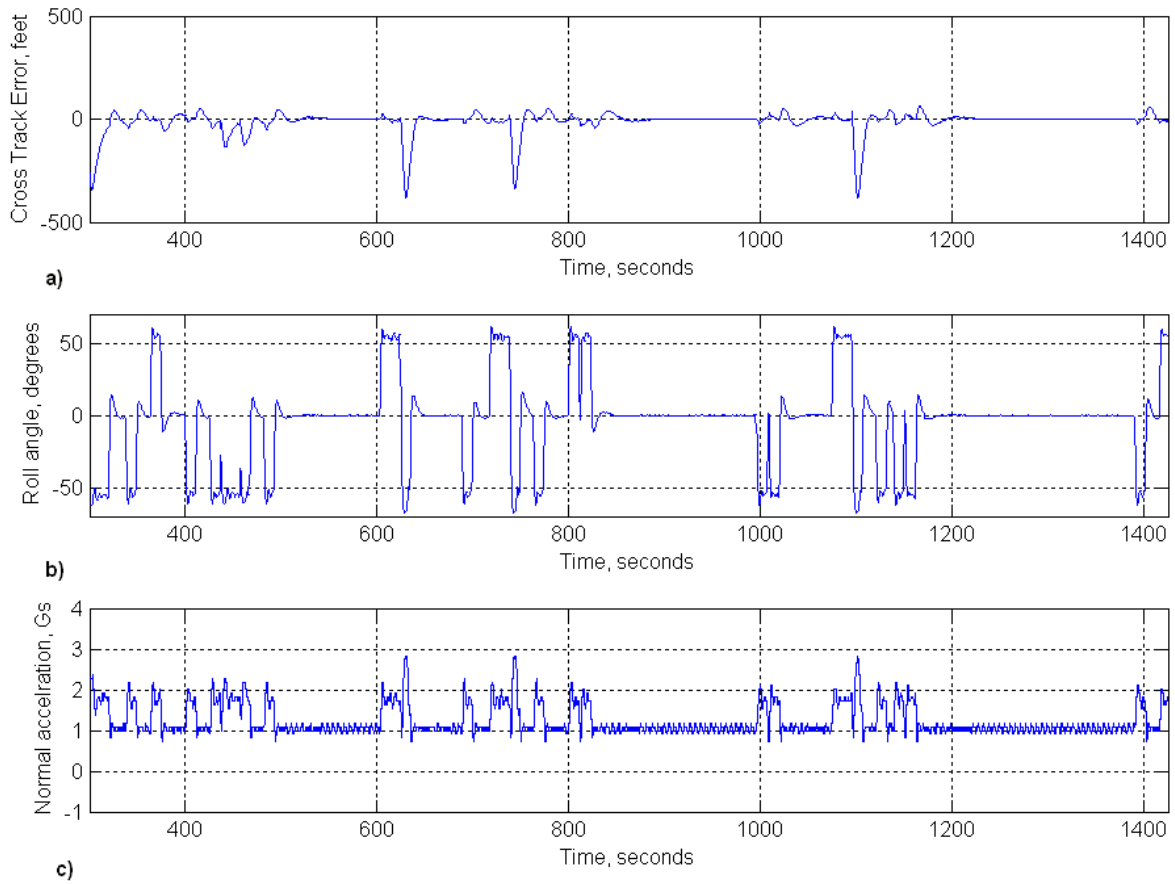


Figure 7. Target lateral control on the 6500 feet MSL FP. *Figure 7 a. Cross track error, Fig. 7 b. Roll attitude angle, Fig. 7 c. Normal acceleration*

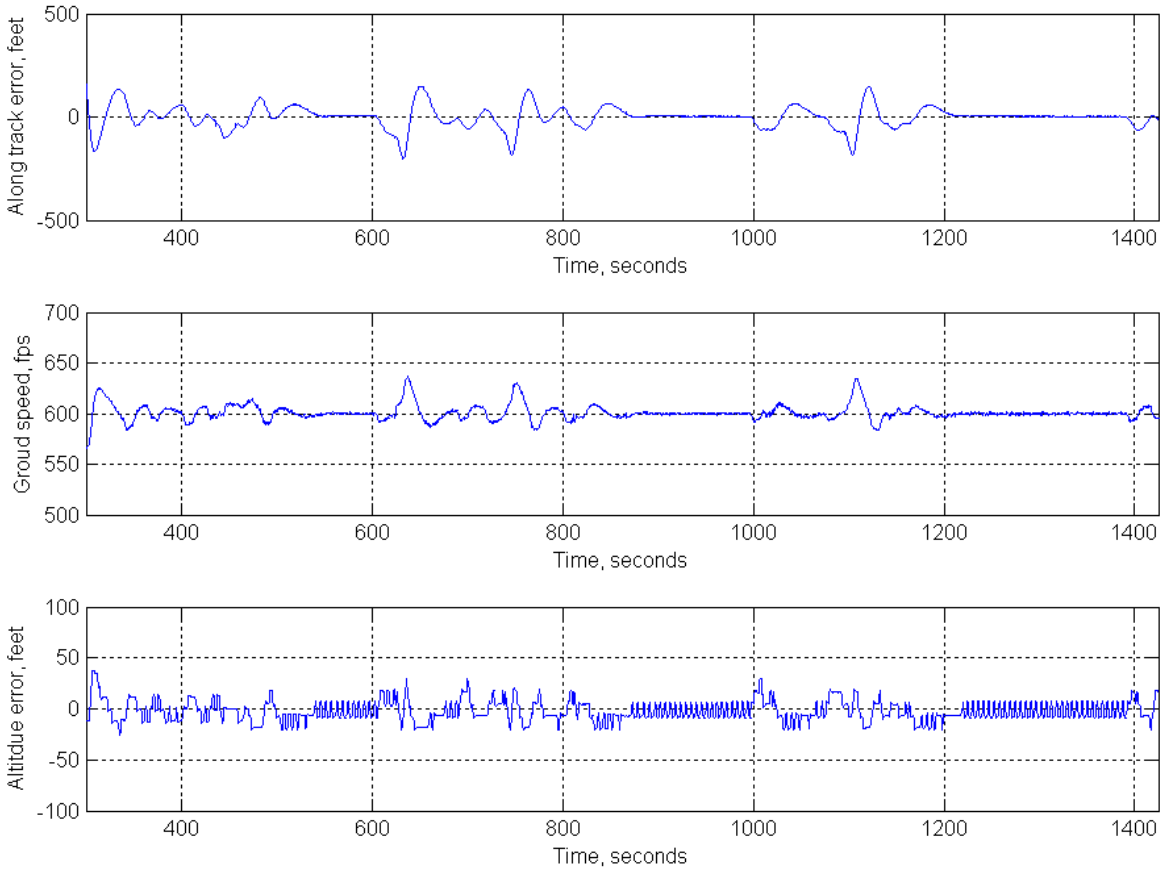
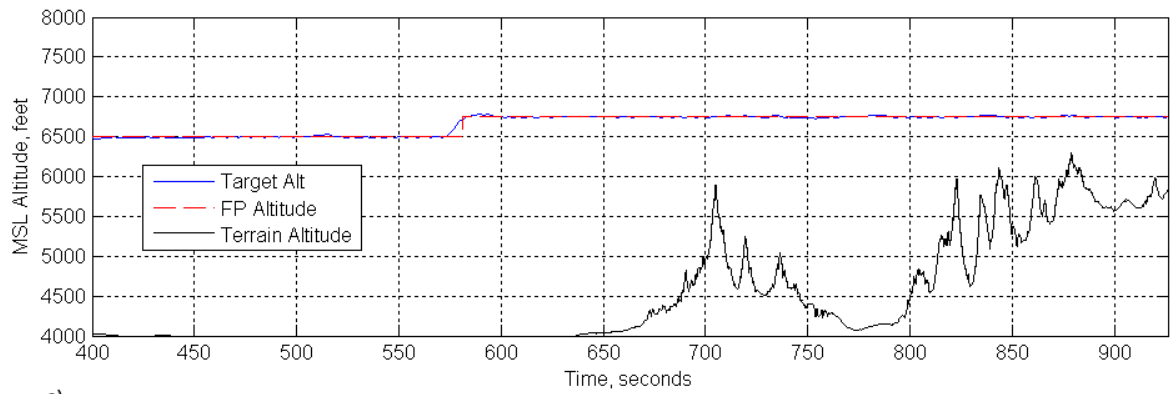


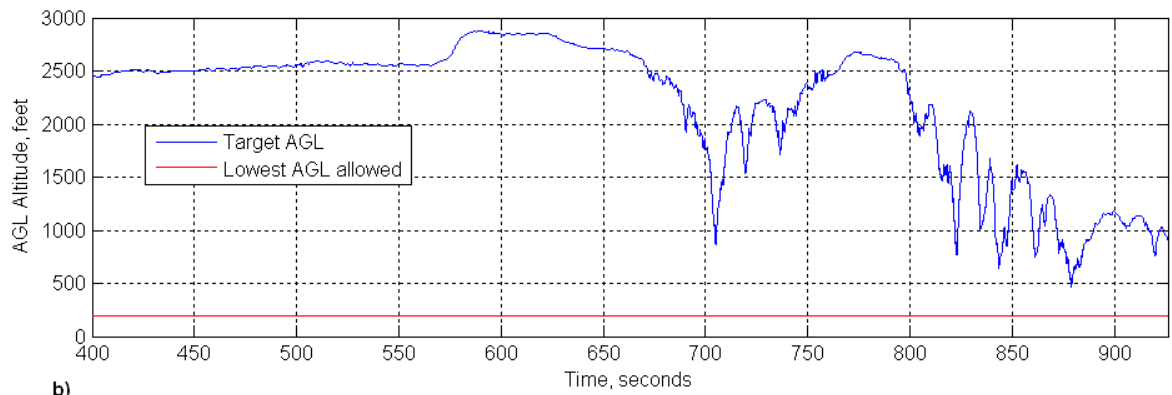
Figure 8. Target longitudinal control on 6500 feet MSL pattern. *Figure 8 a. Along track error, Fig. 8 b. Ground speed, Fig. 8 c. Altitude error*

D. MQM-107D simulation on the 6750 feet MSL pattern

The MQM-107D target was launched and manually flown toward the 6750 feet MSL red pattern (See Fig. 2 and 3). BAH mode was used to level off the target at 6500 feet MSL. SHOT was used to control the speed of the target at 600 ft/s. At point "D" the DFCS operator entered the SEG ASTAR keyboard command defining the goal segment, final target heading, minimum FP altitude AGL (200 feet), and the expected target ground speed in ft/s (600 ft/s). As depicted in Fig. 2a, the RTPP generated a new red pattern at 6500 feet MSL to connect to the pre-existing red pattern at 6750 feet MSL; generated by RTPP prior to target launch. Automatic mode was commanded and the target started following the newly generated red pattern. Figure 10a shows that about 2 ½ minutes later, (at point A in Fig.2a), the drone altitude climbed to 6750 feet when the target reached the 6750 feet MSL pattern. Figure 10a also shows the terrain elevation underneath the target. Figure 10b shows target altitude AGL in feet. Notice that the minimum AGL observed was about 500 feet. Figure 8 shows the terrain elevations in feet close to the flight pattern. Figure 11 also shows that the terrain within 3000 feet from the target is below the flight pattern altitude of 6750 feet. Figure 12 shows cross track error, target roll attitude angle and normal acceleration. The cross track error plot shows that the drone was able to maintain very good cross track control for the entire flight with the exception at t=750 seconds where the target deviated in cross track 350 feet from the flight pattern while on a turn segment. Figure 13 shows time plots of along track error, ground speed and altitude error. The altitude error transient observed at t=575 seconds was due to the commanded target altitude change from 6500 feet 6750 feet as depicted by the red trace in Fig. 10a.



a)



b)

Figure 10. Figure 10 a. Terrain beneath 6750 feet MSL flight pattern, Fig. 10 b. Target AGL and lowest FP AGL allowed by the RTPP

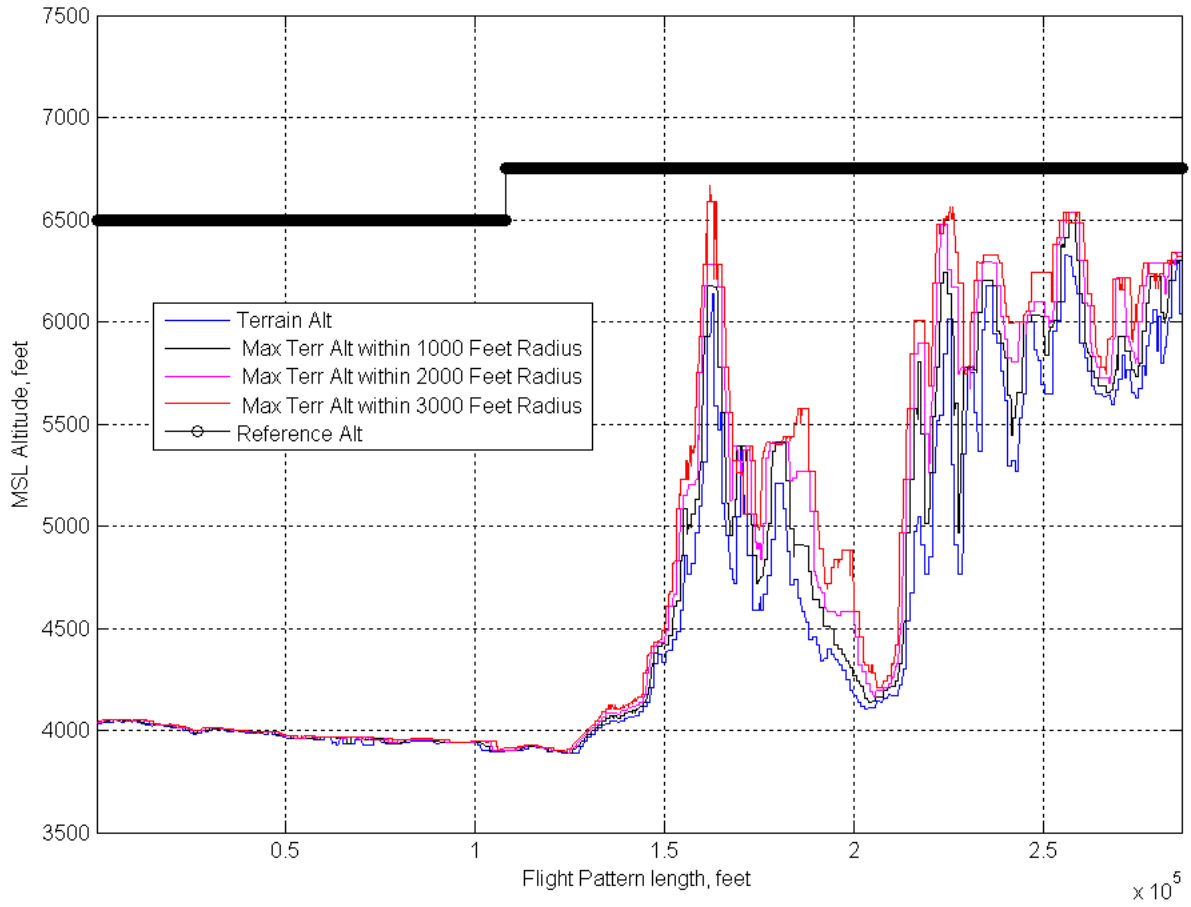


Figure 11. Terrain nearby 6750 feet flight pattern.

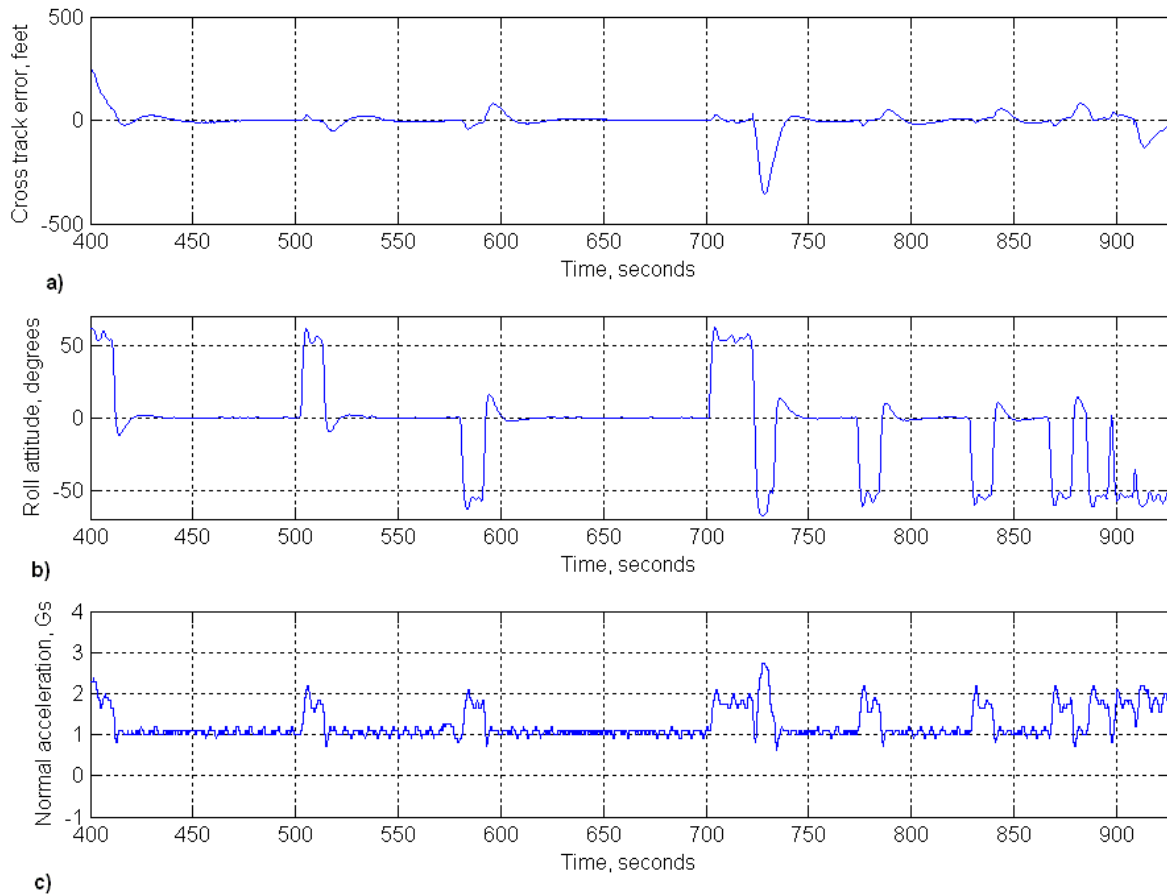


Figure 12. Target lateral control on 6750 feet MSL flight pattern. *Figure 12 a. Cross track error, Fig.12 b. Roll attitude angle, Fig 12 c) Normal acceleration*

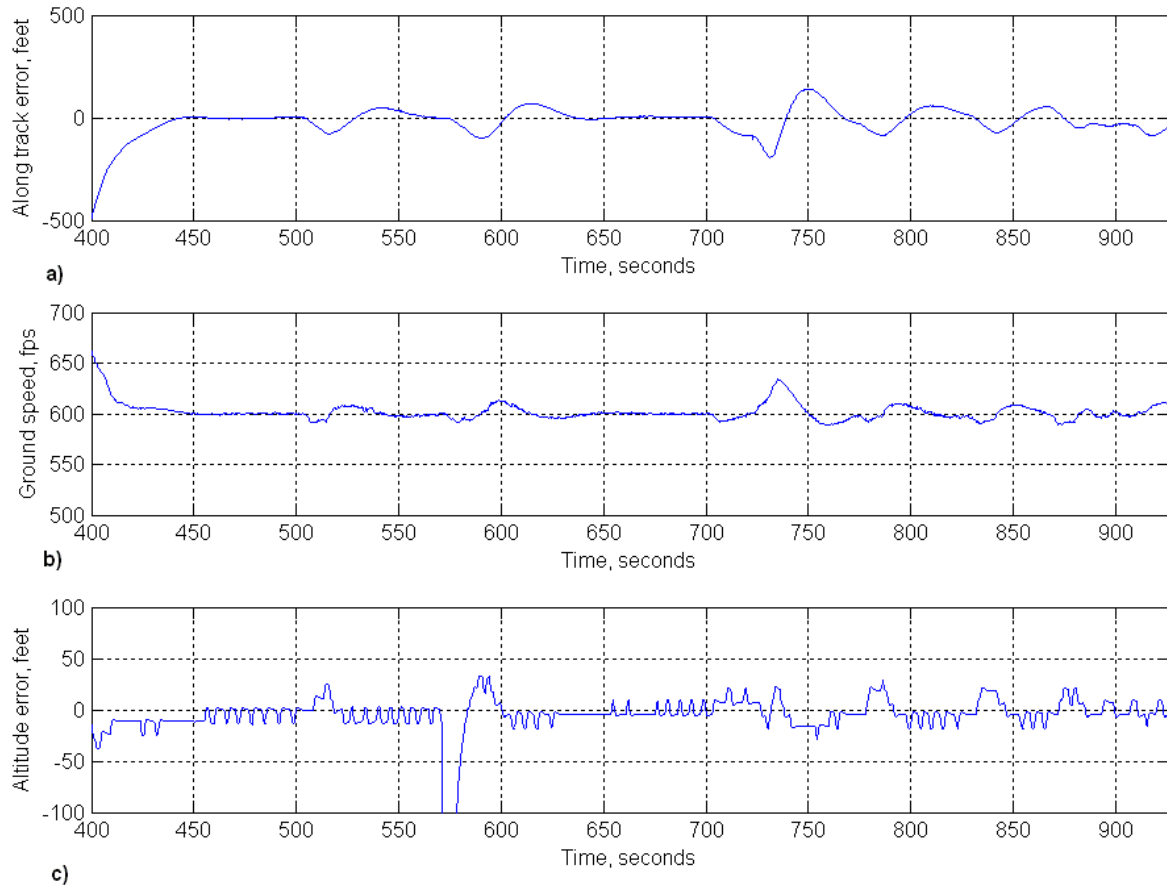
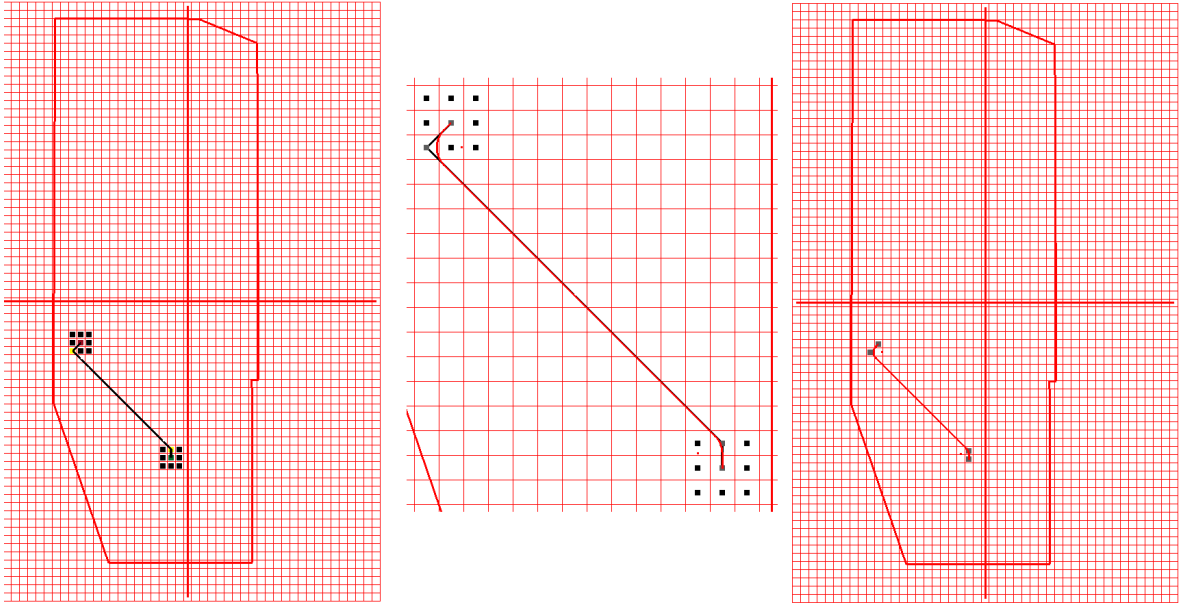


Figure 13. Target longitudinal control on the 6750 feet MSL flight pattern. Fig 13 a. Along track error, Fig 13 b. Ground speed, Fig 13 c. Altitude error

E. MQM-107D simulation on the 7000 feet MSL pattern

The MQM-107D target was launched and manually flown toward the 7000 feet MSL black pattern depicted in Fig. 1 and 2. For this simulation the target was flown all the way to the start of the 7000 feet pattern at point “B”; therefore, there was no need to use the RTPP to automatically generate an acquisition segment to automatically fly the target to the main pattern. Automatic mode was commanded at point “B”. Figure 11 shows that the drone flew at a safe altitude AGL and well above the lowest altitude of 200 feet allowed by the RTPP algorithm. Figure 12 clearly shows that there was no high altitude terrain nearby the target. Figure 13 shows good lateral control of the target. The cross track error plot, Figure 13a shows the time, ($t=40$ seconds), when the drone acquired the flight pattern segment. Figure 14 shows good longitudinal control of the target. The along track error was very close to zero feet, the ground speed was very close to the commanded ground speed of 600 feet per second and the altitude error shows that the drone was at the commanded altitude of 7000 feet MSL.



a) **Figure 14.** RTTP Screen shots. *Figure 14 a. A* path, Fig. 14 b. A* path, waypoints and flight pattern, Fig.14 c.waypoints and flight pattern*

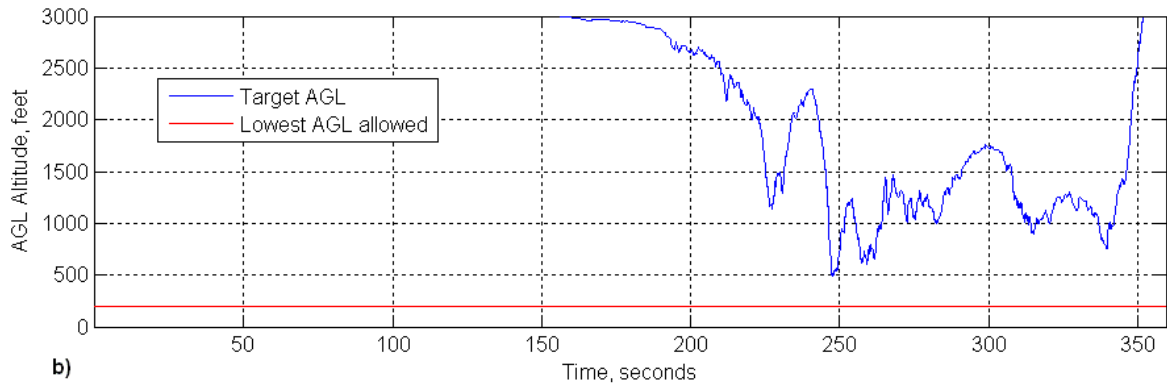
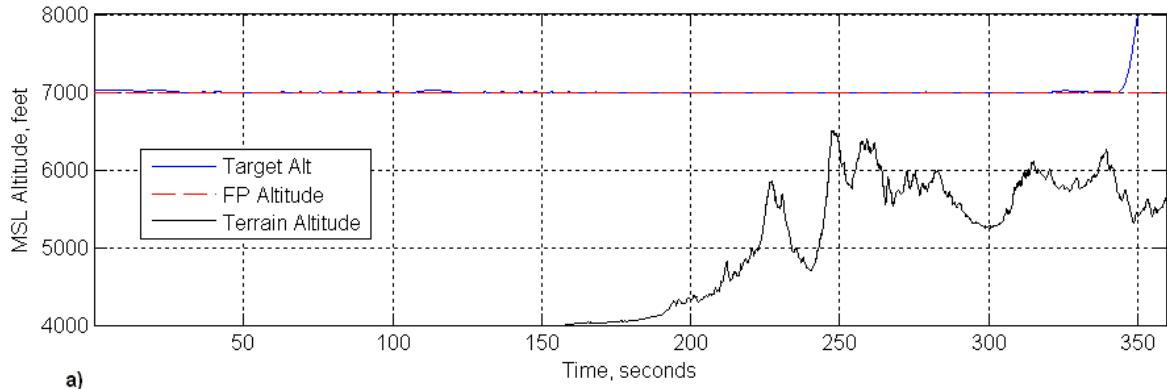


Figure 15. Fig 15 a. Terrain beneath 7000 feet MSL flight pattern, Fig 15 b. Target AGL

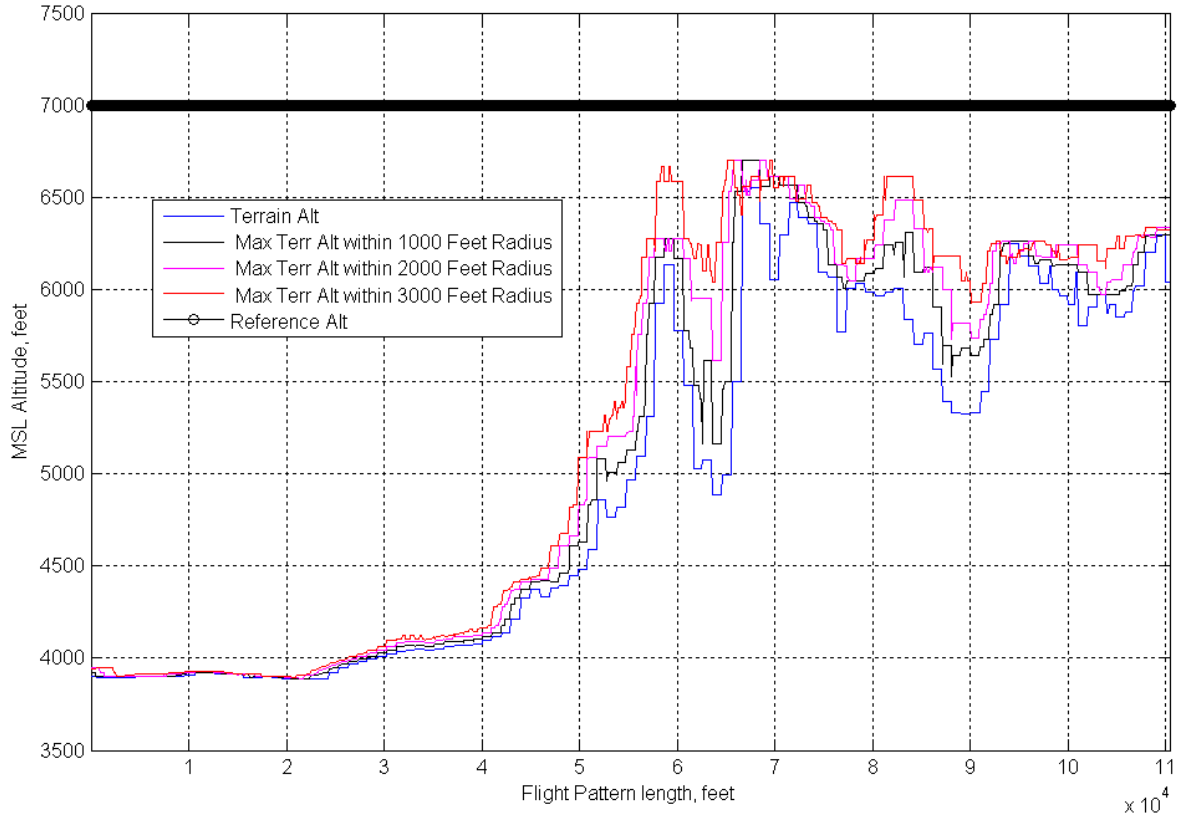


Figure 1. Terrain nearby 7000 feet flight pattern.

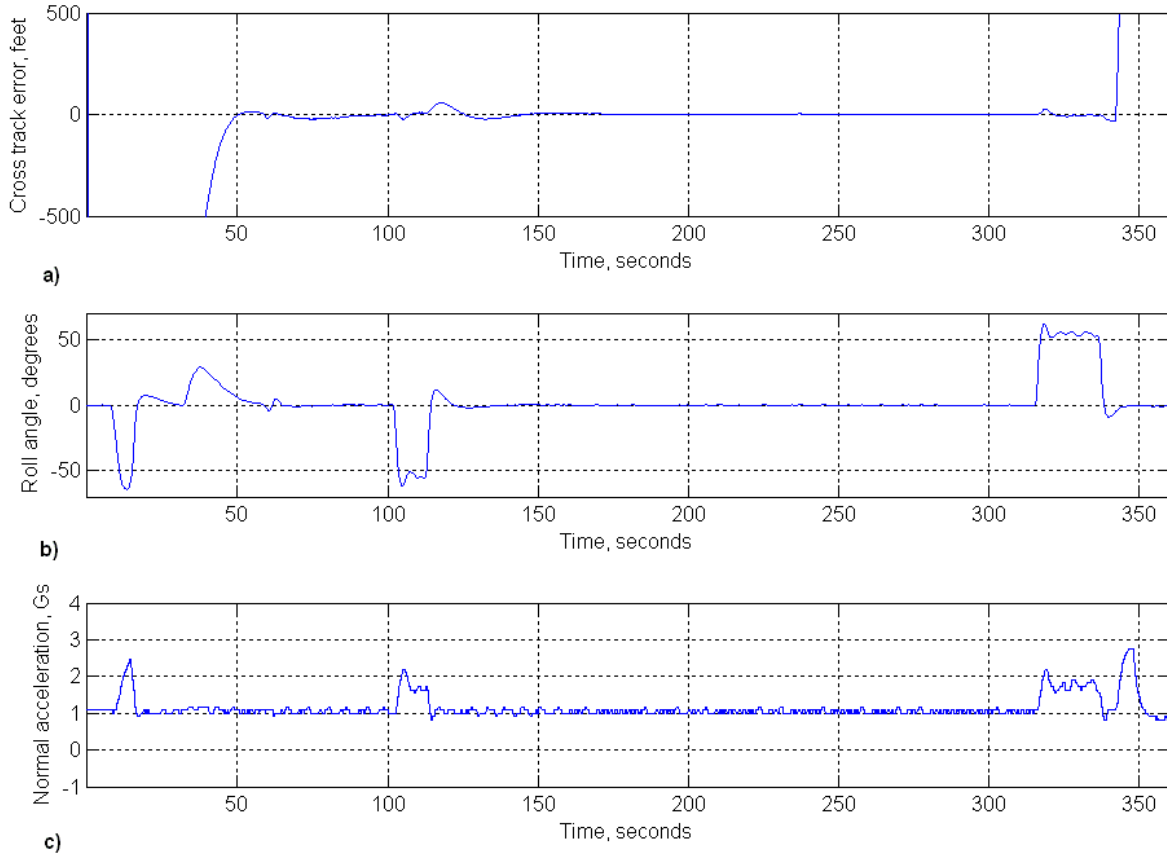


Figure 2. Target lateral control on 7000 feet MSL flight pattern. Fig. 17 a. cross track error, Fig. 17 b. roll attitude angle, Fig 17 c. normal acceleration

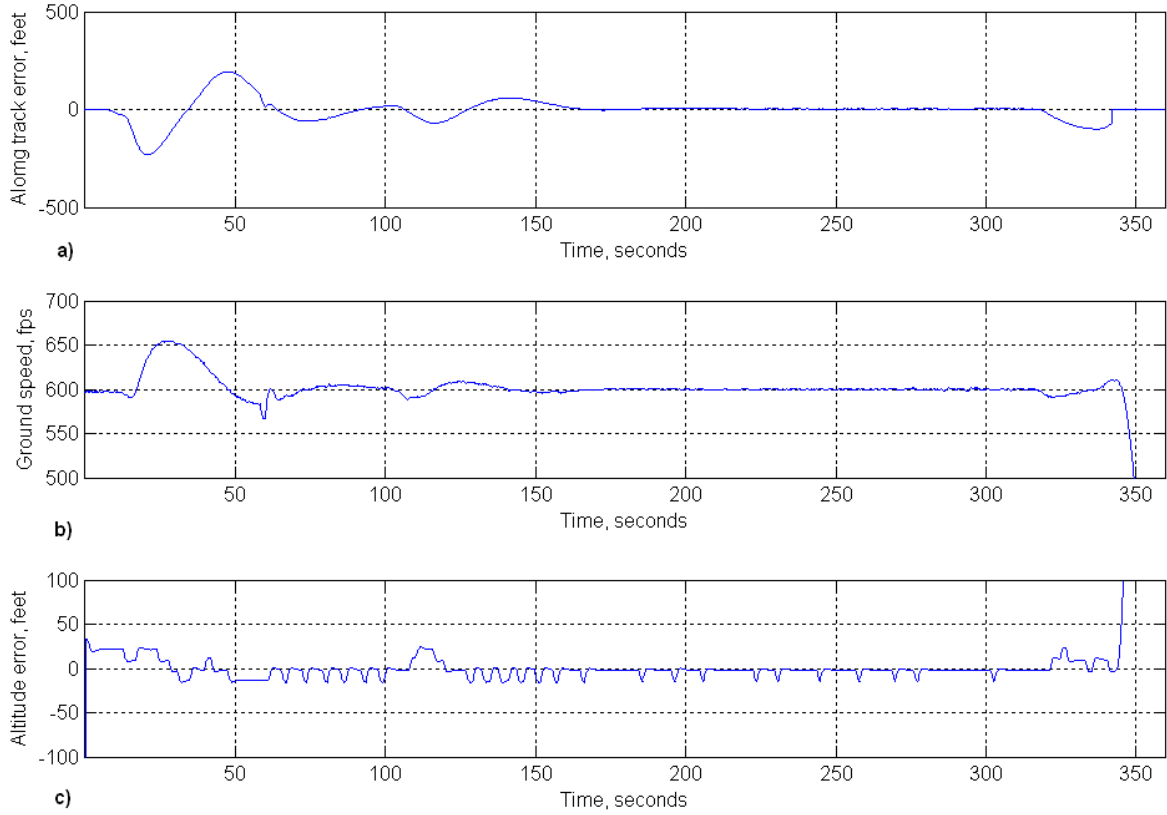


Figure 18. Target longitudinal control on 7000 feet MSL flight pattern. Fig 18 a. along track error, Fig 18 b. ground speed, Fig. 18 c. altitude error

F. Simulation Summary

The 6-DOF simulation tests with the MQM-107D target showed that the RTPP generated flight patterns that are flyable, safe and are ready to be tested with a real MQM-107D target or with the QF-4. The simulator also showed that the flight patterns can be made easier to follow by making the expected ground speed larger than the commanded rabbit speed.

VII. Future Work

As described in the preceding sections the RTPP is on A* algorithm that uses an evaluation function to determine the *best* path between two points on a plane. The evaluation function will be modified in the near future to accommodate new guidance requirements that will be of a great benefit to DOD threat management systems. The following subsections outline some of these potential algorithm improvements:

A. Target Obscuration Requirements

Develop a guidance system that can automatically generate, in real time, a path for the target that is not visible to the combat system that is being evaluated. This will reduce the cost of manually generating target paths using terrain elevation information and sensor (i.e. radar) visibility profiles. This guidance requirement could be accomplished by RTPP by simply using the DTED data, already being processed by RTPP, and available sensor information. Using these data the RTPP cost function could automatically penalize map squares visible to the sensor and could generate a path that does not go over those map locations.

B. Target Synchronization

Develop a target path that not only avoids high terrain and no-fly-zone areas but also has the appropriate length to facilitate controlling the time of arrival of a target to its destination point. The RTPP algorithm will work in conjunction with current DFCS flight path guidance algorithms to control the speed and altitude of the target during its trajectory to its destination point. This could be done with several targets at the same time therefore synchronizing their time of arrivals to pre-defined points on the range.

C. Graphical User Interface

The keyboard commands presently used to control the RTPP functions will be replaced with more modern system user interfaces (e.g. touch screens and GUI).

VIII. Conclusion

The design approach for solving the problems mentioned above was by automating the flight trajectory generation process, and by employing obstacle avoidance and path finding techniques. The use of the obstacle-avoidance and path-finding techniques reduces collision risk while providing a safe route when flying at low elevations over mountainous terrain. Using obstacle-avoidance and path-finding methods with automation of the resulting flight trajectories, solves several problems. First, changes to the flight-trajectory-geometry become safer when done in real-time. Second, specialized knowledge of flight trajectory development is no longer required. Last, mission personnel will have the flexibility of generating flight trajectories during the mission, in real-time, whereas before, the flight trajectories were always created before the flight mission.

References

¹Fausett, L, *Fundamentals of neural networks, architectures, algorithms, and applications*, Prentice Hall, Upper Saddle River, New Jersey, 1994, Ch. 7.

²Barto, A.G., Bradtke, S. J., and Singh, S. P., "Learning to act using real-time dynamic programming," *Artificial Intelligence*, Vol. 72, No. X, 1995, pp. 81 - 138.

³Korf, R.E., "Optimal Path-Finding Algorithms," In Kanal, L. N. and Kumar, V., *Search in Artificial Intelligence*, Springer-Verlag, Berlin, New York, 1988, pp. 222 – 267, 1988

⁴Nilsson, N. J., *Problem-Solving Methods in Artificial Intelligence*, McGraw-Hill Inc., New York, 1971. Ch. 3

⁵Mitchell, J.S.B., "An Algorithmic Approach to Some Problems in Terrain Navigation", *Artificial Intelligence*, Vol. 37, No. 37, pp. 171-201

⁶Capozzi, B., D. Rathbun, S. Kragelund, and A. Pongpunwattana, "An evolution path planning algorithm for autonomous motion of a UAV through uncertain environments," *IEEE Proceedings: The 21st Digital Avionics Systems Conference*, 2002. Proceedings.pp. 8D2-1 - 8D2-12 vol.2.

⁷Yao-hong Q., P. Quan, and Y. Jian-guo, "Flight path planning of UAV based on Heuristically Search and Genetic Algorithms," IEEE 32nd Annual Conference of Industrial Electronics Society, 6-10 Nov. 2005, pp. Digital Object Identifier 10.1109/IECON.2005.1568876.

⁸Russel, S. and Norvig, P., *Artificial Intelligence A Modern Approach*, 2nd Ed., Prentice Hall, Upper Saddle River, New Jersey, 2003, pp. 96 – 104.

⁹U.S.G.S., "Seamless Data Distribution System," Earth Resources Observations and Science, URL: <http://seamless.usgs.gov/> [cited 30 April 30, 2007].

¹⁰U.S.G.S., "National Geospatial-Intelligence Data," NGA: DoD World Geodetic System 1984, URL: http://earth-info.nga.mil/GandG/publications/tr8350.2/tr8350_2.html [cited 30 April 30, 2007].

¹¹Raney, J. T., and K. D. Rehm, "Precision Location, Navigation and Guidance Using DME Techniques," IEEE Position Location and Navigation Symposium, 1976.