A new model for optimal TF/TA flight path design problem

R. Zardashti
Faculty of Aerospace Engineering
Amir Kabir University of Technology
Tehran, Iran

M. Bagherian
bagheri1@khayam.ut.ac.ir
School of Mathematics, Statistics and Computer Science
University College of Science
University of Tehran, Tehran, Iran

ABSTRACT

This paper focuses on the three dimensional flight path planning for a UAV on a low altitude terrain following/terrain avoidance mission. Using an approximate grid-based discretisation scheme, we transform the continuous optimisation problem into a search problem over a finite network, and apply a variant of the shortest-path algorithm to this problem. In other words using the three dimensional terrain information, three dimensional flight path from a starting point to an end point, minimising a cost function and regarding the kinematics constraints of the UAV is calculated. A network flow model is constructed based on the digital terrain elevation data (DTED) and a layered network is obtained. The cost function for each arc is defined as the length of the arc, then a constrained shortest path algorithm which considers the kinematics and the altitude constraints of the UAV is used to obtain the best route. Moreover the important performance parameters of the UAV are discussed. Finally a new algorithm is proposed to smooth the path in order to reduce the workload of the autopilot and control system of the UAV. The numeric results are presented to verify the capability of the procedure to generate admissible route in minimum possible time in comparison to the previous procedures. So this algorithm is potentially suited for using in online systems.

NOMENCLATURE

(i,j)	a node of the network
$[(i,j),(k,j+1)]$	an arc of the network
ARMS	automated routing and maintenance system
AS	admissible set
CRG	candidate route generator
CLOAR	common low observable auto-router
U_1	control on local rate of climb
U_2	control on local turn rate
V	cruise velocity
DTED	digital terrain elevation data
g	gravity
JASSM	Joint Air-Surface Standoff Missile
n	load factor
ROC_{max}	maximum rate of climb
J	performance measure
$pp(i,j)$	the distance label of node (i,j)
$d[(i,j),(k,j+1)]$	the cost associated with arc $[(i,j),(k,j+1)]$
$x_{i,j,k,j+1}$	the amount of flow on arc $[(i,j),(k,j+1)]$
$\mathbf{X_0}$	the initial state
$\mathbf{X_f}$	the goal state
$\mathbf{X}$	the state vector

Paper No. 3366. Manuscript received 3 November 2008, revised version received 22 February 2009 accepted 30 March 2009.

https://doi.org/10.1017/S0001924000002979 Published online by Cambridge University Press

J_{fuel}	total fuel consumption
R	turn radius
γ	local rate of climb
$\Delta\psi$	local turn angle
$\dot{\psi}$	local turn rate

1.0 INTRODUCTION

A central task in aircraft mission planning is selecting the best route for the mission. Route planning is an important problem for a number of diverse applications including intelligent transportation systems, space applications, autonomous robotics and military guidance and navigation systems[1-9]. Past approaches to automated route planning have usually formulated the problem as a matter of numeric cost minimisation to be solved by methods like dynamic programming[10], genetic algorithm[11], numerical procedures[6,7,12-14] or geometric algorithms like *A** search[16,17]. Most geometrical route planning algorithms can be considered either graph-based or grid-based, each with their own associated advantages and disadvantages. Graph-based approaches are usually very accurate, but since some missions require several constraints on the resultant path, such as not exceeding minimum turn radius and threshold comfort level and/or workload of the autopilot, if these constraints are involved, these procedures faced difficulties such as long convergence time, even exponential[18]. Grid-based techniques use a digitised grid consisting of square cells of equal size, representing the environment in which the route planning is performed. The size of the environment is $m \times n$ grid cells. The best route is chosen as that route from start to goal which has the lowest total cost usually measured in terms of fuel usage and survivability. *A**-based algorithms are variants of the shortest path algorithm. Grid-based approaches converge in real-time but have difficulties with route constraints. Most military applications use DTED as input to their route planner. A cost estimation is assigned to a particular cell corresponding to the cost incurred by traveling through that particular region. Then a route is planned from a given start grid cell location to a goal grid sell location with the least total cost. CRG is based on a best-path first, tree search which uses *A** algorithm to find the least-cost path from a given initial point on a graph to a goal point on a graph. The geographic map area of interest over which the UAV is to fly is overlaid with a graph with a certain pre-defined density of squares. The intent of the graph is to assign a 'cost' to every possible movement between two cells in the graph. This score must reflect a complex set of objectives for the overall route. Therefore, the computation of this set of costs makes up a significant portion of the CRG algorithm. Given the graph and associated costs, and the start and end points for a UAV route, the CRG algorithm analyses candidate routes that are made up of smaller segments that cross edges of the graph squares or diagonals of the graph squares. This analysis is performed according to a segment cost minimisation function. CRG attempts to algorithmically obtain the lowest total route cost. The route that best minimises overall cost in the graph space is assumed to be the most optimal route for the real-world UAV[9]. This approach is strongly based on the cost map which should be produced and stored; and the cost map production is a very time consuming task. In dynamic programming a local cost is assigned to each link of the grid which spans the area of operation. This assumes that the cost of flying over an area is independent of the path by which the UAV reached the area, which is not always true. Both dynamic programming and *A** search techniques suffer from their relatively slow execution. Graph search methods are a standard tool for shortest-path problems. However, existing work does not address problems with dynamic aspects, namely when accelerations are taken into account as part of the problem constraints and cost[19]. Graph search methods are closely related to dynamic programming, but employ the special structure of the shortest-path problem to enhance computational efficiency. Some researchers have formulated the optimal path planning problem as an optimal control problem which is equivalent to a two-point boundary value problem[6,7]. As discussed in Ref. 12 this approach is problematic, since the adjoin equations are frequently nonlinear and obtaining them is unwieldy for complex dynamics. Moreover, convergence of the corresponding iterative solution procedure is often very sensitive to the accuracy of the initial guess of the adjoin equation; and there is no physical interpretation of the adjoin variables to provide an intuition into what constitutes a good initial guess. Many authors have investigated alternative approaches to the solution of the optimal control problem[13,20,21]. However, their approaches suffer from long convergence time. The interested reader is refereed to[3,4] for more discussion about different route planning approaches.

Some systems have been developed for this purpose too. The common low observable auto-router (CLOAR) is used on UAVs which are considered 'low observable' or stealth, such as F-117A, the B-2 and the Joint Air-Surface Standoff Missile (JASSM)[22,23]. CLOAR is a classified product, but is purported to determine figure of merit costs as it routes and determine least threat cost route based on inputs. CLOAR is deployed as a module within the Air Force Mission Support System, which is used for mission planning on the aforementioned weapons systems. CLOAR route planning times are on the order of hours. Thus it cannot be used as a real-time in-flight mission planner. The automated routing and maintenance system (ARMS), is another route planner that computes sortie effectiveness as part of the preplanning process, and analyses data after a sortie has been flown[3]. Like CLOAR, ARMS does not have real-time in-flight capabilities.

In this paper a new model of the route planning problem is introduced which is based on a layered network flow structure, constructed on the DTED. The present work aims to develop a 3D trajectory optimization method which is of global nature, yet computationally feasible, using the basic tool of network flows algorithms. Such a model has not been considered previously whereas numerical simulations show the effectiveness and robustness of this method for optimal path generation. The rest of the paper is organised as follows. In Section 2 the flight mechanics formulation of the problem is presented. In Section 3 the mathematical model and the network flow structure of it are constructed. In Section 4 the optimality conditions are expressed and the constrained shortest path algorithm based on the optimality conditions and altitude constraints of the UAV are proposed. Additional kinematics constraints are imposed to the route in Section 5. An algorithm is proposed to reduce the extra altitude changes of the path to decrease the workload of the autopilot in Section 6, and numerical simulations are presented in Section 7. Section 8 ends the paper with conclusion.

2.0 FLIGHT MECHANICS FORMULATION

2.1 Model

We start with a simplified, kinematics model of a UAV flying in a three-dimensional airspace. The UAV is considered as a point in 3D space, and its translational and angular states at time t are defined by the configuration vector $\mathbf{X}(t) = (x(t), y(t), z(t), \gamma(t), \psi(t))^T \in G$. The set G is defined as $\Re^3 \times (\gamma_{min}, \gamma_{max}) \times (-\pi, +\pi)$. The co-ordinates x, y and z are taken relative to an inertial reference frame. The extremal values γ_{min} and γ_{max} are derived from the flight model, detailed in the next subsection. The flight speed $V(t)$ will be taken constant. We note that during climbs, even with throttle compensation, the UAV can lose roughly 20% of its velocity, and during descends can gain up to 20% increased velocity. However, by proposing an altitude smoothing algorithm we minimise these effects. Wind speed data may also be used to modifying V according to the heading angle $\psi(t)$. But we shall not proceed further here with a precise modelling of these effects. We assume instantaneous control of the angular

https://doi.org/10.1017/S0001924000002979 Published online by Cambridge University Press

rates, so that the control signal $U(t)$ coincides with $(\gamma,\dot{\psi})$. The state equations $\dot{\mathbf{X}}(t) = f(\mathbf{X}(t), \mathbf{U}(t))$, $0 \le t \le t_f$ may now be written as

$$\begin{aligned}
\dot{x}(t) &= V(t)\,\mathrm{Cos}\,\gamma(t)\,\mathrm{Cos}\psi(t)\\
\dot{y}(t) &= V(t)\,\mathrm{Cos}\,\gamma(t)\,\mathrm{Sin}\psi(t)\\
\dot{z}(t) &= V(t)\,\mathrm{Sin}\,\gamma(t)\\
\dot{\gamma}(t) &= U_1(t)\\
\dot{\psi}(t) &= U_2(t)
\end{aligned} \quad \ldots(1)$$

The control and state variables will be further constrained by the flight model. Given an initial state X_0 and the goal state X_f, feasible trajectories are continuous solution paths $X(t)$ that start at $X(0) = X_0$ and terminate at $X(t_f) = X_f$. We shall be interested in optimal trajectories that minimise a cost functional to be defined below.

2.2 Flight model constraints

UAV performance is subject to physical constraints that affect its maneuverability. Additional restrictions may be applied to ensure autopilot comfort. To determine the flight model constraints, horizontal and vertical channels are considered independently. Rate of climb and turn rate are used to limit controls U_1 and U_2 respectively as follows:

$$\gamma_{max} = \mathrm{Sin}^{-1}(ROC_{max}/V) \quad \ldots(2)$$

$$\dot{\psi}_{max} = 57\cdot 3 \times \frac{V}{R_{min}} \quad \ldots(3)$$

Where ROC_{max} and $\dot{\psi}_{max}$ represent the maximum rate of climb and maximum turn rate respectively, and $R_{min} = \dfrac{V^2}{g\sqrt{n_{max}{}^2 - 1}}$ is the minimum turn radius of the UAV in which g and n_{max} are the gravity and the maximum continues load factor of the UAV respectively. We obtain the following constraints on the rate of climb and turn rate:

$$\gamma \le \gamma_{max} \quad \ldots(4)$$

$$\dot{\psi} \le \dot{\psi}_{max} \quad \ldots(5)$$

2.3 The cost function

The flight duration t_f is an obvious performance measure. It may be expressed as a unit integral over the flight interval:

$$J = \int_0^{t_f} 1 \mathrm{d}t \quad \ldots(6)$$

More generally, the fuel consumption rate may be approximated as a function of the flight configuration, leading to the cost component J_{fuel} which evaluates the total fuel consumption. We note that in a typical application, the fuel consumption is approximately proportional to the flight time, and in our experiments we only consider the latter. Since we have assumed the constant cruise velocity for the UAV, minimising the flight duration is equivalent to minimising the length of the flight path. So we find constrained shortest path as the optimal path.

We are interested in optimal trajectories that minimise the above cost functional. Numeric solutions of continuous dynamic programming methods may require multiple passes over the entire state space. For the specific case of the shortest-path problem (which corresponds to a special case of the problem) more efficient alternatives are offered by appropriate discretisation schemes coupled with an efficient constrained shortest path algorithm as described below.

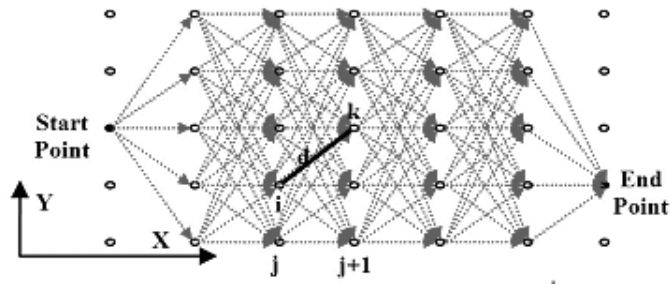


Figure 1. The layered network constructed on DTED.

3.0 PROBLEM MODELLING AND DEFINITIONS

We discretise the model over a special region which includes spatial position, and formulate the trajectory optimisation problem as a constrained shortest-path problem over a layered network flow structure. In this paper we use a digitised grid consisting of rectangle cells of equal size to represent the route planning environment. A DTED which consists of an $m \times n$ grid is used as input to our route planning algorithm. Each grid point is of the form (x,y,z) representing the position co-ordinates with respect to some predefined inertial reference frame. A network flow structure is constructed based on the grid of DTED. We refer to each of the grid points as a node of the network. Since the DTED grid is regular, the nodes can be classified as a layered network with n layers and each layer contains m nodes. Since finding the shortest possible path is desired, UAV should fly always toward the goal. Thus each node in the jth layer is connected to all nodes in the $(j+1)$th layer via n arcs, so arc $[(i,j), (k, j+1)]$ connects the ith node in the jth layer , as the tail of the arc, to the kth node in the $(j+1)$th layer, as the head of the arc. (see Fig. 1) This form is appropriate for the application of the shortest-path algorithm, which is an efficient single-pass algorithm. In essence, the algorithm proceeds from the start (or initial) node and iteratively develops the front of equi-distant nodes. Thus, at each step nodes of one layer are tagged with their optimal distance label, until a target node is reached. As will be seen, the sequential structure of this scheme will allow us to introduce some important modifications to the algorithm.

We associate with each arc a cost which is defined as the spatial distance between its tail and head as follows

$$\mathrm{d}\left[(i,j),(k,j+1)\right] = \sqrt{\left(x_{i,j} - x_{k,j+1}\right)^2 + \left(y_{i,j} - y_{k,j+1}\right)^2 + \left(z_{i,j} - z_{k,j+1}\right)^2} \quad \ldots(7)$$

Note that all the arcs connect nodes of two consecutive layers. We use this property to propose a suitable variant of the shortest path algorithm to speedup finding the optimal route. We can formulate the problem mathematically as follows:

$$\begin{aligned}
&\mathit{Minimize}: \sum_{\{[(i,j),(k,j+1)]\in A\}} d\left[(i,j),(k,j+1)\right] \times x_{i,j,k,j+1}\\
&\mathit{Subject\ to}: \sum_{\{(k,j+1):[(i,j),(k,j+1)]\in A\}} x_{i,j,k,j+1}\\
&\quad - \sum_{\{(k,j+1):[(i,j),(k,j+1)]\in A\}} x_{k,j+1,i,j}\\
&\quad = \begin{cases} m\times n - 1 & for\ (i,j) = Start\ Point\\ -1 & for\ (i,j) \ne Start\ Point\end{cases}\\
&\quad x_{i,j,k,j+1} \ge 0 \quad for\ all \quad [(i,j),(k,j+1)] \in A
\end{aligned} \quad \ldots(8)$$

Where $x_{i,j,k,j+1}$ represents the amount of flow which passes through arc $[(i,j),(k,j+1)]$. After calculating the shortest path if arc $[(i,j),(k,j+1)]$ is on the shortest path we have $x_{i,j,k,j+1} > 0$, otherwise we have $x_{i,j,k,j+1} = 0$.

https://doi.org/10.1017/S0001924000002979 Published online by Cambridge University Press

4.0 THE SHORTEST PATH ALGORITHM

To solve Equation (8), we have to find $x_{i,j,k,j+1}$ for all $[(i,j),(k,j+1)] \in A$, to minimise the objective function of Equation (8). We have proposed a variant of the shortest path algorithm which is suitable for layered networks. This algorithm uses the optimality conditions of the shortest path problem to find the optimal route. To express the optimality conditions, we associate with each node of the network a distance label which represents the shortest distance from the start node to that node. We refer to distance label of node (i,j) as $pp(i,j)$.

4.1 The optimality conditions

Theorem. The necessary and sufficient condition for the optimality of the distance labels is that for all nodes (i,j) and $(k,j+1)$ which are connected via arc $[(i,j),(k,j+1)]$ we have

$$pp(k,j+1) \le pp(i,j) + \mathrm{d}\left[(i,j),(k,j+1)\right] \quad \ldots (9)$$

Moreover arc $[(i,j),(k,j+1)]$ is on the optimal path if and only if

$$pp(k,j+1) = pp(i,j) + \mathrm{d}\left[(i,j),(k,j+1)\right] \quad \ldots (10)$$

Proof. The interested reader can find the proof of the theorem in Ref. 19.

4.2 Computing the optimal distance labels

To initialise the algorithm, set the distance label of the start node to zero and set the distance label of all other nodes to infinity. Now by applying a forward scheme the distance labels of nodes in each layer can be computed using the optimality conditions and the distance labels of the previous layer. Suppose that the optimal distance labels of the jth layer have been computed. Then for each node k in $(j+1)$th layer set

$$pp(k,j+1) = \min_i \left\{ pp(i,j) + \mathrm{d}\left[(i,j),(k,j+1)\right] \right\} \quad \ldots (11)$$

According to Equations (9) and (10) $pp(k,j+1)$ is the optimal distance label of node $(k,j+1)$. Following this scheme, the optimal distance label of all nodes including the goal node can be computed. Note that since the computed distance labels satisfy the optimality conditions (Equations 9 and 10), they represent the length of the shortest path from the start node to those nodes. Specially the distance label of the goal node represents the length of the shortest path from the start node to the goal node.

4.3 Determining the optimal route

After computing the optimal distance label of the goal node, we can determine the optimal route as follows. Let (l,m) be the goal node of the network and $pp(l,m)$ be its optimal distance label. Let $(r,m-1)$ be some node in $(m-1)$th layer for which

$$pp(l,m) = pp(r,m-1) + \mathrm{d}\left[(r,m-1),(l,m)\right] \quad \ldots (12)$$

Then, according to Equation (12), arc $[(r,m-1),(l,m)]$ is on the optimal path. By applying a backward scheme we can find arc $[(u,m-2),(r,m-1)]$ in $(m-2)$th layer for which $pp(r,m-1) = pp(u,m-2) + \mathrm{d}[(u,m-2),(r,m-1)]$. Continuing this backward scheme we finally reach to the start node.

5.0 IMPOSING ADDITIONAL CONSTRAINTS

Since military UAVs are restricted to fly in low altitudes to avoid detection by threats and to use terrain masking, we can also impose altitude constraint to the generated path. For this purpose it is sufficient to do as follows. First a threshold value for maximum flyable altitude for the UAV should be specified. Then while computing the optimal distance labels, if the altitude of some node is greater than the threshold value, by setting its optimal distance label to infinity, the designed path doesn't pass through regions with altitudes greater than the threshold value. Using this technique, we can express the threat regions as areas with the altitudes greater than the threshold value. This causes the generated path avoid from threat regions automatically. This leads to a path which is composed of segments of straight lines. However, because of the kinematics constraints of the UAV, this path is not tractable and has to be modified to meet the constraints such as local turn rate and local rate of climb constraints. On the other words this path should be smoothed somehow to become tractable by the UAV. In the next section, we impose some constraints on the path which leads to a much more tractable flight path.

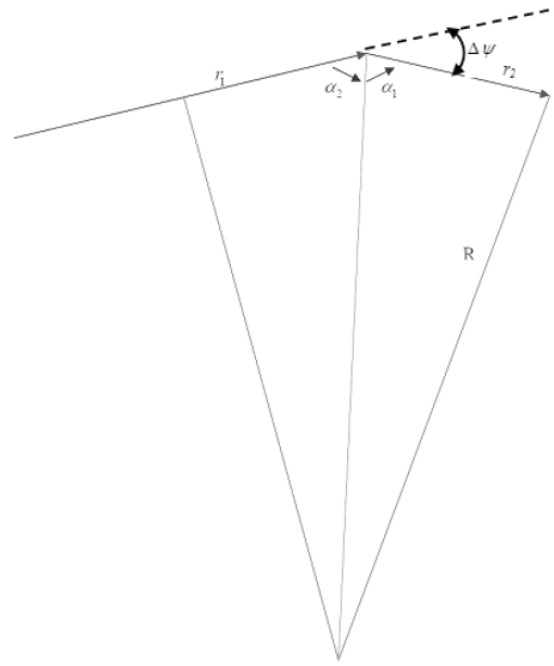


Figure 2. The geometric representation of the local turn angle in xy-plane.

If vehicle kinematics constraints were not a concern, the generated path would be feasible and optimal, but due to kinematics constraints, it is necessary to convert a series of path segments to a feasible trajectory which is tractable by the UAV. Generating a feasible trajectory reduces to generating a set of feasible transitions, or turns, between successive path segments in xy-plane and generating feasible climbs in xz-plane. Let $\Delta\psi_{max}$ be the maximum local turn angle of the UAV at some node. Since the generated path consists of straight line segments, the UAV should turn at the connecting points of the two successive straight line segments. In order to compute $\Delta\psi$, let $(x_{k,j-2}, y_{k,j-2}, z_{kj-2})$, $(x_{i,j-1}, y_{i,j-1}, z_{ij-1})$, and $(x_{k',j}, y_{k',j}, z_{k',j})$ be three nodes in three consecutive layers of the network. To compute the angle which is obtained by connecting these points in xy-plane, we define two vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ in xy- plane as:

$$\mathbf{r}_1 = (x_{i,j-1} - x_{k,j-2}, y_{i,j-1} - y_{k,j-2}) \quad \ldots (13)$$

and

$$\mathbf{r}_2 = (x_{k',j} - x_{i,j-1}, y_{k',j} - y_{i,j-1}) \quad \ldots (14)$$

https://doi.org/10.1017/S0001924000002979 Published online by Cambridge University Press

Thus the local turn angle at point $(x_{i,j-1}, y_{i,j-1}, z_{ij-1})$ can be calculated as

$$\Delta\psi = \text{Arccos}\frac{\boldsymbol{r}_1.\boldsymbol{r}_2}{\|\boldsymbol{r}_1\|.\|\boldsymbol{r}_2\|} \quad \ldots (15)$$

Moreover we require that $|\dot{\psi}| \leq \dot{\psi}_{max}$, owing to the UAV constraints. Consider again $(x_{kj-2}, y_{kj-2}, z_{kj-2})$, $(x_{ij-1}, y_{ij-1}, z_{ij-1})$ and $(x_{k'j}, y_{k'j}, z_{k'j})$ as three nodes in three consecutive layers of the network; Regarding $\mathbf{r}_1$ and $\mathbf{r}_2$ as defined by Equations (13) and (14) and are shown in Fig. 13. Let $r = min\{\mathbf{r}_1,\mathbf{r}_2\}$.We have $\alpha_1 = \alpha_2$ and $\alpha_1 + \alpha_2 = \pi - \Delta\psi$, thus $\alpha_1 = \frac{\pi - \Delta\psi}{2}$; Moreover $\text{Cos}(\alpha_1) = \frac{r}{R}$, hence $R = \frac{r}{\text{Cos}\left(\frac{\pi}{2} - \frac{\Delta\psi}{2}\right)}$, and we can obtain $\dot{\psi}$ using Equation (3).Therefore we can impose the local turn rate constraint $|\dot{\psi}| \leq \dot{\psi}_{max}$, to the path planning algorithm by setting the distance label of node $(x_{i,j-1}, y_{i,j-1}, z_{ij-1})$ to infinity if Equation (3) does not satisfied at this node .

Another important performance parameter of the UAV is the rate of climb γ, which can be calculated as follows. Let $(x_{kj-2}, y_{kj-2}, z_{kj-2})$, $(x_{ij-1}, y_{ij-1}, z_{ij-1})$ and $(x_{k'j}, y_{k'j}, z_{k'j})$ be three nodes in three consecutive layers of the network. Define two vectors $\mathbf{r}_3$ and $\mathbf{r}_4$ in *xz*- plane as:

$$\boldsymbol{r}_3 = (x_{i,j-1} - x_{k,j-2}, z_{i,j-1} - z_{k,j-2}) \quad \ldots (16)$$

and

$$\boldsymbol{r}_4 = (x_{k',j} - x_{i,j-1}, z_{k',j} - z_{i,j-1}) \quad \ldots (17)$$

Thus the local rate of climb at point $(x_{i,j-1}, y_{i,j-1}, z_{ij-1})$ can be calculated as

$$\gamma = \text{Arccos}\frac{\boldsymbol{r}_3.\boldsymbol{r}_4}{\|\boldsymbol{r}_3\|.\|\boldsymbol{r}_4\|} \quad \ldots (18)$$

If the maximum rate of climb is γ_{max} (which can be calculated by Equation (4), then by computing γ at any point $(x_{i,j-1}, y_{i,j-1}, z_{ij-1})$ and comparing it by γ_{max}, we can restrict the path to pass through feasible rate of climbs by setting the distance label of node $(x_{i,j-1}, y_{i,j-1}, z_{ij-1})$ to infinity, if Equation (4) does not satisfied at this node.

We call this modified algorithm the constrained shortest path algorithm.

6.0 DESIGNING THE ALTITUDE SMOOTHING ALGORITHM

As we have mentioned above, during climbs, even with throttle compensation, the UAV can lose roughly 20% of its velocity, and during descends can gain up to 20% increased velocity. Since we want to have an approximately constant cruise velocity, to minimize these effects the altitude smoothing algorithm is proposed. An additional concern for trajectory planning is the autopilot workload. During the flight, the autopilot is preoccupied with numerous tasks on top of the need to follow the outlined trajectory. Having a rapidly varying trajectory requires the autopilot to constantly focus on the task of following it, especially for low flight over irregular terrain. We shall therefore rely on the riding qualities cost to regulate the smoothness of the planned trajectory.

In other words many altitude changes will increase the workload of the autopilot and is not comfortable for control system of the autopilot. So it is necessary to smooth the path somehow to eliminate unwanted altitude changes. Moreover, the UAV need not always stay close to the ground. Instead it is sufficient to stay low enough so as not to be visible to the surrounding threats. This means that there may be regions along the route where the UAV can fly considerably above the terrain following command height and still avoid detection. The advantage of doing this as we mentioned above, is that the velocity changes and the work load of the autopilot are both reduced.

To this aim we have proposed an algorithm which we called it altitude smoothing algorithm. Consider the constrained optimal path generated by the constrained shortest path algorithm. The following algorithm smoothes the altitude changes of the UAV.

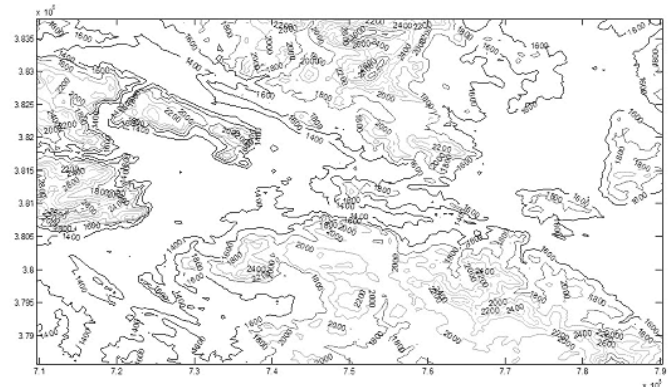


Figure 3. The digitised model of the area loaded in MATLAB software.

6.1 Altitude smoothing algorithm

Consider the constrained optimal path in *xz* plane.

1. Determine a threshold value for admissible altitude changes.
2. Determine the ascending and descending intervals of the path.
3. Define an admissible set (AS) consisting of admissible nodes of the path and add the start node and the goal node to AS.
4. Compute the altitude difference between the first node and the last node of each interval obtained in Stage 2.
5. For each interval, if the altitude difference computed in Stage 4 is greater than the threshold value of Stage 1, add the last node of the interval to AS.
6. The desired optimal path passes through the points of AS.

The proposed path avoids the altitude changes less than the specified threshold value and a smoothed path with less workload of the autopilot is obtained which is more tractable by the UAV.

7.0 NUMERICAL SIMULATIONS

To verify the performance of our optimal path planning algorithm, numerical simulations are performed. The proposed algorithm is compared with an optimal control procedure[25]. The algorithmic engine is a Microsoft Visual FORTRAN® program, and the graphics engine is MathWorks MATLAB®. The algorithm is run with $\dot{\psi}_{max}$ = 7deg/sec, γ_{max} = 7deg and V = 230ms^{-1}. We have used a real DEM as input to our algorithm. Figure 3 shows the digitised model of this area loaded in MATLAB® software. The resolution of this map is 34,020 nodes, (m = 210 and n = 162) with Δx = 250m, Δy = 500m.

Figure 4 shows the unconstrained and constrained paths generated by the algorithm. In all figures unconstrained paths are shown by red color and kinematics constrained paths are shown by blue colour. Figure 5 shows the unconstrained and constrained paths in *xy*-plane and Fig. 6 shows the constrained path in *xz*-plane (red), the altitude smoothed path (cyan) and the more smoothed path (blue).

The execution time is approximately 85 seconds, based on the hardware which is used. To compare our procedure with the optimal control procedure[8] we have used the same region with identical start and end points and threat region. The run time of our algorithm is much faster. It takes about 20 seconds while in optimal control

https://doi.org/10.1017/S0001924000002979 Published online by Cambridge University Press

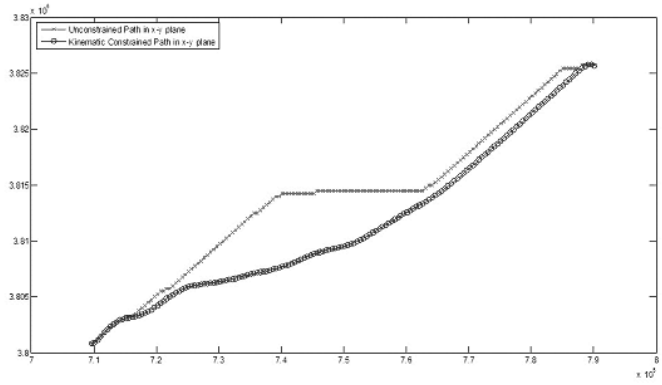


Figure 4. The path without kinematics constraints (red) and with kinematics constraints (blue).

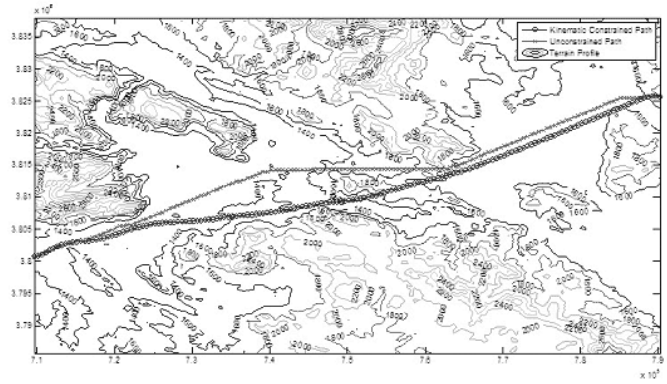


Figure 5. Generated paths in *xy*-plane.

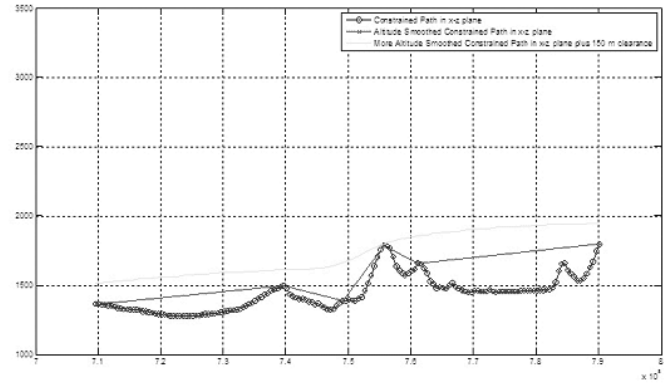


Figure 6. Generated paths in *xz*-plane.

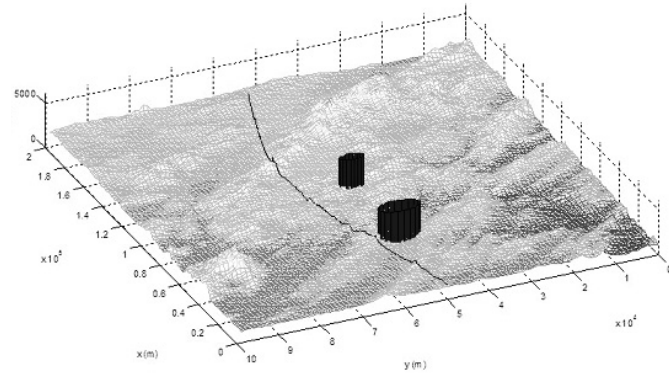


Figure 7. The path generated by optimal control procedure.

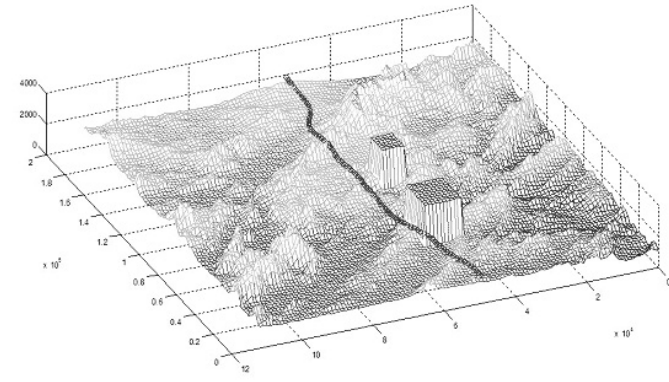


Figure 8. The path generated by constrained shortest path algorithm.

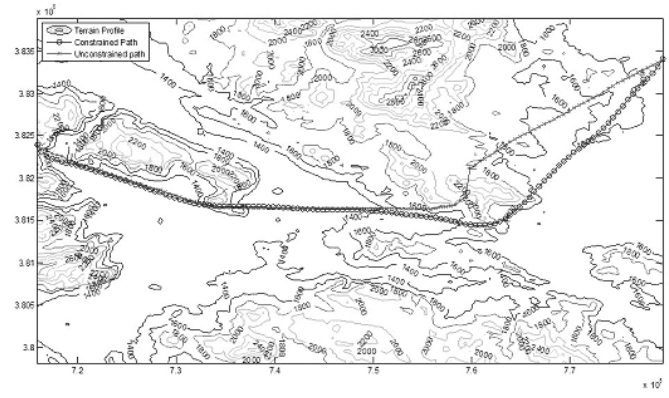


Figure 9. The unconstrained kinematics (red) and kinematics constrained (blue) paths.

procedure every iteration takes about 150 seconds. Also the optimal control procedure is sensitive to the initial heading angle. Different initial heading angles lead to paths with different miss distances. So the best initial heading angle should be determined by repeating the algorithm with different heading angles. The best heading angle is obtained as the heading angle which has the minimum miss distance to the goal node. Moreover it is very difficult to impose the maximum flyable altitude constraint in optimal control formulation with constant velocity assumption. The optimal control procedure needs an interpolated surface of the region which is a time consuming task. Figures 7 and 8 show the generated paths.

We repeated the simulations with another DEM with the resolution of 52,224 nodes, (m = 408 and n = 128) and Δx = 100m, Δy = 500m and with different start and end points. The choice of the grid resolution is a decisive factor in the complexity and accuracy of the resultant path as is shown by numerical simulations. Figure 9 shows the unconstrained and constrained paths generated by the algorithm. Figure 10 shows the unconstrained and constrained paths in *xy*-plane and Fig. 11 shows the constrained path in *xz*-plane (red), the altitude smoothed path (cyan) and the more smoothed path (blue).

If we have a threat region on the DTED, as shown in Figs 12 and 13, the generated paths avoids this region automatically. The execution time of the algorithm for this resolution is about 500 seconds, based on the hardware which is used. The numerical simulations show that the more nodes present in the network, the more accurate path with shorter length is found; and the more execution time is required. Thus to get an optimal resolution leads to

https://doi.org/10.1017/S0001924000002979 Published online by Cambridge University Press

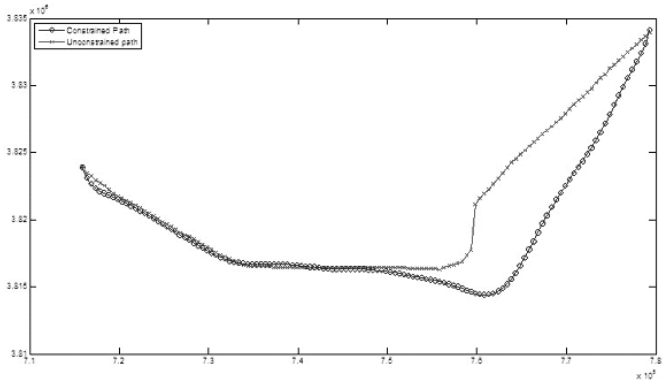


Figure 10. The obtained paths in *xy*-plane.

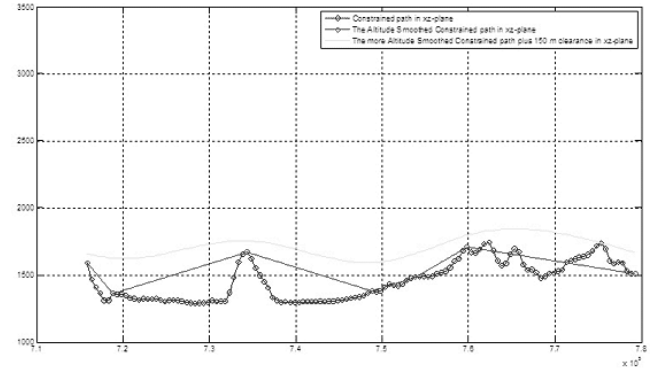


Figure 11. The obtained paths in *xz*-plane.

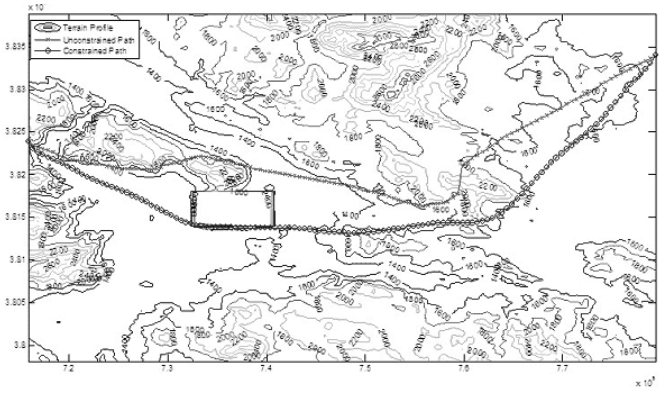


Figure 12. The path without kinematics constraints (red) and the path with constraints (blue) avoiding a rectangle threat region.

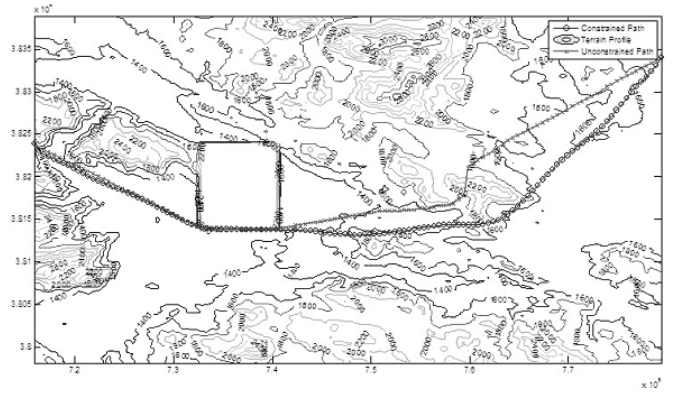


Figure 13. The path without kinematics constraints (red) and the path with constraints (blue) avoiding a larger rectangle threat region.

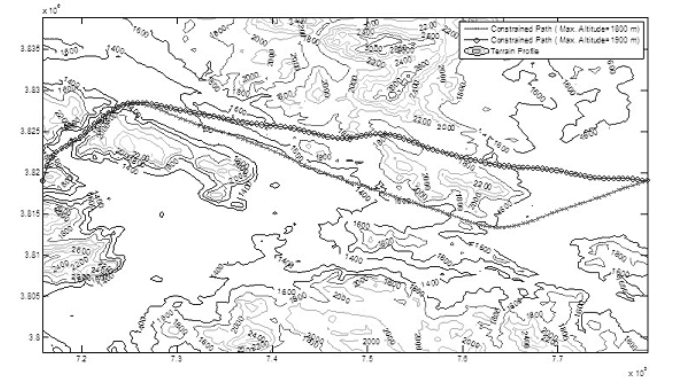


Figure 14. The path with all constraints.

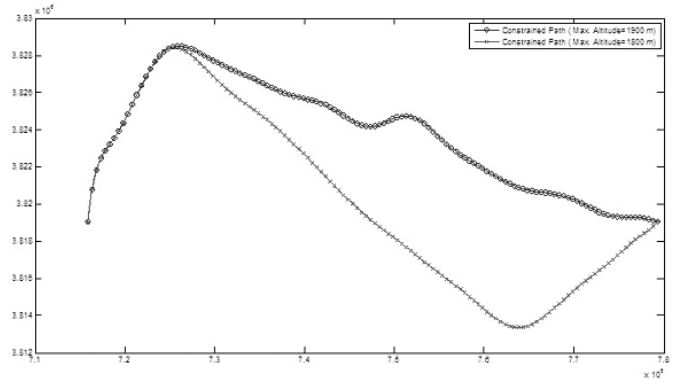


Figure 15. The resultant constrained path in *xy*-plane.

a tradeoff between the length of the obtained path and the running time of the algorithm.

We repeated the simulations with the same DEM with the resolution of 26,112 nodes, (m = 204 and n = 128) and Δx = 200m, Δy = 500m and with different start and end points and different maximum flyable altitudes. Figures 14 and 15 show the resultant paths. Figures 16 and 17 show the control commands provided by the algorithm.

To verify the performance parameters $a_z = \dot{\gamma} \times V$, $a_y = \dot{\psi} \times V$ and $R = \frac{V}{\dot{\psi}}$ these parameters can be computed and compared with their admissible values. For our example $a_z < 4g$, $a_y < 4g$ and R_{min} = 1,800m, which are in the acceptable ranges of our UAV.

8.0 CONCLUSION

In this paper a new model was proposed to formulate the optimal TF/TA flight path design problem. This model is based on a layered network flow structure which is constructed on the DTED of the region on which the path have to be designed. According to the optimality conditions of the shortest path problem in network flows theory a variant of the constrained shortest path algorithm is proposed to solve the problem regarding the kinematics and the altitude constraints of the UAV. Numerical simulations show the effectiveness and robustness of this method for optimal path generation. The advantages of our procedure can be referred as follows. First of all, the procedure produces three dimensional flight path, while the geometric procedures of Refs 18 and 24-26

https://doi.org/10.1017/S0001924000002979 Published online by Cambridge University Press

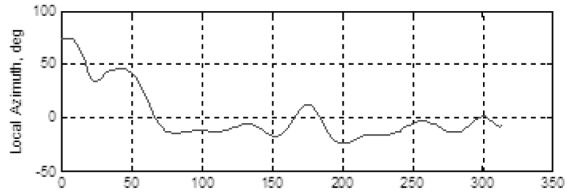


Figure 16. The control command for local azimuth for the path of Fig. 10.

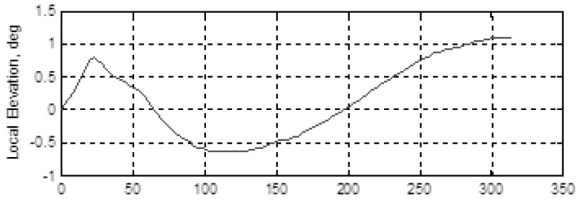


Figure 17. The control command for local elevation for the path of Fig. 10.

produce two dimensional flight paths in a much longer time. The second advantage is that our procedure doesn't need to calculate and store the cost map as a data bank in comparison to Ref. 9 which is based on this very time consuming task. As the last advantage, our procedure is able to find the exact solution of the problem, while the genetic algorithm[2] only finds the near optimal solution; and the optimal control procedure[13] has a large miss distance from goal. The numerical simulations show that to obtain an optimal resolution leads to a tradeoff between the accuracy of the path and the execution time of the algorithm, since the more nodes present in the network leads to a shorter path and more execution time.

ACKNOWLEDGEMENTS

The authors appreciate the valuable comments provided by the anonymous referee which led to this improved version of the paper.

REFERENCES

1. AL-HASAN, S. and VACHTSEVANOS, G. Intelligent route planning for fast autonomous vehicles operating in a large natural terrain, *Robotics and Autonomous Systems*, 2002, **40**, pp 1–24.
2. BODENHORN, C., GALKOWSKI, P., STILES, P., SZCZERBA, R. and GLICKSTEIN, I. Personalizing onboard route re-planning for recon, attack, and special operations missions, September 1997, American Helicopter Society Conference (Avionics and Crew Systems Technical Specialists Conference).
3. HWANG, Y. and AHUJA, N. Gross motion planning — a survey, *ACM Computing Surveys*, September 1992, **24**, (3) , pp 219-291.
4. LATOMBE, J. *Robot Motion Planning*, 1991, Kluwer, Boston, MA.
5. SZCZERBA, R., CHEN, D. and UHRAN, I. A framed-quadtree approach for determining Euclidean shortest paths in a 2-D environment, 1997, *IEEE Transactions on Robotics and Automation*, October 1997, **13**, (5), pp 668-681.
6. SHARMA, T., WILLIAMS, P., BILL, C. and EBERHARD, A. Optimal three dimensional aircraft terrain following and collision avoidance, 2007, *ANZIAM J*, 2005, EMAC, pp C695-C711.
7. VINCENT, T.L. and GRANTHAM, W.J. *Nonlinear and Optimal Control Systems*, 1999, John Wiley and Sons.
8. TWIGG, S., CALISE, A. and JOHNSON, E. 3D trajectory optimization for terrain following and terrain masking, 2006, AIAA Guidance, Navigation, and Control Conference and Exhibition, Keystone, CO, 21-24 August 2006.
9. www.virtualacquisitionshowcase.com/docs/2008/Scientific1-Brief.pdf
10. BELLMAN, R. *Dynamic Programming*, 1962, Princeton University Press, Princeton, NJ.
11. Cruise missile auto routing system, 2008, Transitioning Technology to the Fleet, 2-4 June 2008, Hyatt Regency, Crystal City, VA.
12. BETTS, J.T. and HUFFMAN, W.P. Path constrained trajectory optimization using sparse sequential quadratic programming, *J Guidance, Control and Dynamics*, January-February 1993, **16**, (1), pp 59-68.
13. HALL, R. Path planning and autonomous navigation for use in computer generated forces, 2007, Scientific Report, Swedish Defense Research Agency.
14. MALAEK, S.M.B. and KOSARI, A.R. A novel minimum time trajectory planning in terrain following flight, 2003, IEEE Aerospace and Electrical System Conference, **8**, Big Sky, MT.
15. SZCZERBA, R. New Cell Decomposition Techniques for Planning Optimal Paths, 1996, PhD dissertation, University of Notre Dame, Notre Dame, IN.
16. HART, P. NILSSON, N. and RAPHAEL, B. A Formal basis for the heuristic determination of minimum cost paths, *IEEE Transactions on System, Science and Cybernetics*, July 1968, **4**, (2), pp 100-107.
17. RIPPEL, E., BAR-GILL, A and SHINKIN, N. Fast graph-search algorithms for general aviation flight trajectory generation, 2004, Technion — Israel Institute of Technology, Haifa, Israel, 24 May 2004.
18. CORMEN, T., LEISERSON, C. and RIVEST, R. *Introduction to Algorithms*, 1990, McGraw Hill, New York.
19. RAVINDRA, K. AHUJA, T., MAGNANTI, L. and JAMES, B.O. *Network flows, Theory, Algorithms and Applications*, 1993, Prentice Hall Englewood Cliffs.
20. ENRIGHT, P.J. and CONWAY, B.A. Discrete approximations to optimal trajectories using direct transcription and nonlinear programming, *J Guidance, Control and Dynamics*, July-August 1993, **15**, (4), pp 994-1002.
21. LU, P. Inverse dynamics approach to trajectory optimization for an aerospace plane, *J Guidance, Control and Dynamics*, July-August 1993, **16**, (4), pp 726-732.
22. MITCHEL, E..F. The use of preprocessed cruise missile data for strategic planning, 1996, Department of Defense, United States Strategic Command.
23. RODRIQUES, L. Defense acquisitions — achieving B-2A bomber operational requirements, July 1999, GAO/NSIAD-99-97 Report.
24. HARGRAVES, C.R. and PARIS, S.W. Direct trajectory optimization using nonlinear programming and collocation. *J Guidance, Control and Dynamics*, July-August 1987, **10**, (4), pp 338-342.
25. HELGASON, R.V., KENNINGTON, J.L. and LEWIS, K.R. Cruise missile mission planning: a heuristic algorithm for automatic path generation, *J Heuristics*, September 2001, **7**, (5), pp 473-494.
26. AKRAM, I.M., PASHA, A. and IQBAL, N. Optimal path planner for autonomous vehicles, *Proceedings of World Academy of Science, Engineering and Technology*, January 2005, **3**, pp 134-137.

https://doi.org/10.1017/S0001924000002979 Published online by Cambridge University Press