Radar-Assisted Collision Avoidance/Guidance Strategy for Planar Flight

B. AJITH KUMAR
D. GHOSE
Indian Institute of Science

We propose a radar-assisted collision avoidance/guidance strategy (RACAGS) for flight vehicles on low-altitude missions. The task of obstacle avoidance and guidance are integrated in a single collision avoidance/guidance strategy. The avionic aids and computational requirements are modest as the strategy mainly depends on range-map and inertial system information. The strategy is first implemented in a planar scenario and then extended to three-dimensional and nominal trajectory following flight scenarios. Several simulation studies are presented for illustration.

Manuscript received June 12, 1999; revised August 21, 2000; released for publication September 30, 2000.

IEEE Log No. T-AES/37/1/02918.

Refereeing of this contribution was handled by P. K. Willett.

Authors' current addresses: B. A. Kumar, Department of Electrical Engineering, College of Engineering, Trivandrum 695 016, Kerala, India; D. Ghose, Department of Aerospace Engineering, Indian Institute of Science, Bangalore 560 012, India, E-mail: (dghose@aero.iisc.ernet.in).

I. INTRODUCTION

Obstacle avoidance is a fundamental requirement in the trajectory planning of aerospace vehicles flying at low altitudes. In the modern battlefield environment, helicopters and cruise missiles have to fly at low altitudes to avoid easy detection by ground radars. In military surveillance applications low altitude flight is necessary to detect objects camouflaged by vegetation and other natural obstructions. In many civilian applications, such as agricultural insecticide spraying operations or aerial photography, the vehicle has to fly at low altitudes close to the ground. In the low-altitude flight regime, the vehicle is likely to encounter both natural and man-made obstacles. These obstructions give rise to challenging problems to low-altitude flight missions. Apart from helicopters and rotorcraft [1–3], and low-flying cruise missiles [4], the other types of aerial vehicles that require obstacle avoidance capability are some categories of unmanned aerial vehicles (UAV) [5], microaerial vehicles (MAV) [6], and certain types of low-flying aircraft [7].

Advances in computational capacity, sensor capability, and signal processing have produced a variety of avionic aids for use in the low-altitude flight regime. But many aerial vehicles, especially MAVs and UAVs, cannot carry too many avionic aids due to space constraint and cost. All these factors establish the need for developing collision avoidance strategies that require low computational effort and that depend on minimal avionic aids. Collision avoidance systems built based upon such strategies will serve as simple but effective aids for the pilot and can even be used for automating the task of collision avoidance in low altitude pilotless aerial vehicles.

For low-altitude missions, apart from the requirement of collision avoidance, a basic mission objective for the vehicle is to reach a destination or goal point. Another related task is that of following a predefined strategy as closely as possible. This is the task of the pilot in a piloted vehicle and of a guidance system in an unmanned vehicle. Thus, the collision-avoidance system should be considered as a part of the guidance system which generates maneuver commands necessary to guide the vehicle towards its goal while avoiding obstacles on its path.

There are three recognized modes of flights at low altitude [1–3]: *low-level flight*, *contour flight*, and *nap-of-the-Earth flight* (NOE). *Low-level flight* is generally carried out above the obstacles at a constant altitude. *Contour flight* is carried out at low altitude, conforming generally to the contours of the terrain. NOE flight is carried out close to the surface of the Earth with the distinguishing feature that mainly lateral maneuvers are used to avoid obstacles. It is the NOE flight that is closest to the planar flight regime that we address here.

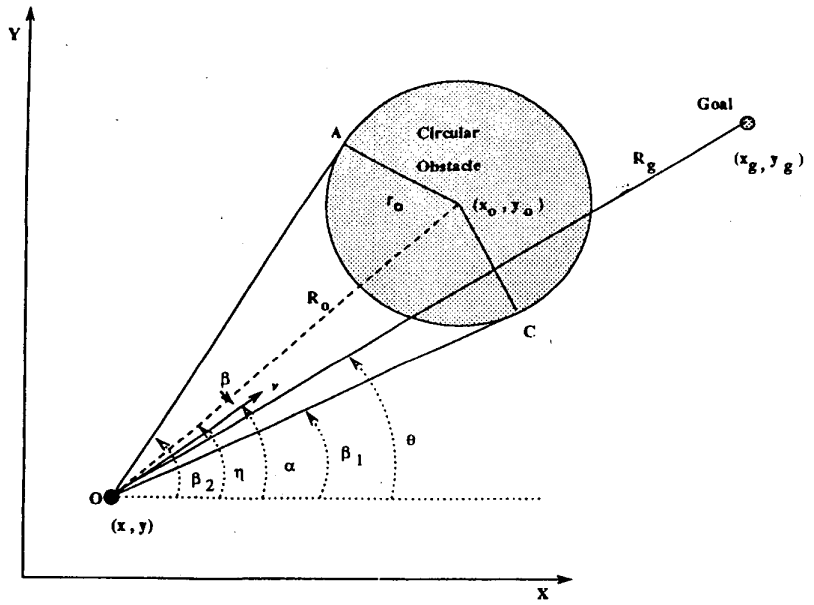


Fig. 1. Geometry of cone of radar returns.

In general, the collision avoidance/guidance problem can be decomposed into three functional levels [2, 3]: *far-field*, *mid-field*, and *near-field*. Far-field mission planning selects goals and intermediate waypoints. Mid-field mission planning produces the nominal optimal trajectory. The near-field mission treats the nominal trajectory as a reference input to the innermost guidance loop and avoids obstacles by using information provided by the obstacle-detection subsystem. It is the near-field mission that is of interest to us in this work.

In many applications it may be necessary for the flight vehicle to pass through (or pass in close proximity to) several intermediate goal points (also known as waypoints), or even follow a given nominal trajectory as accurately as possible. For example, a vehicle on a surveillance mission may be required to take photographs of the terrain at certain prespecified points on its trajectory and then proceed to its goal point. Another related application is the requirement to follow as closely as possible a given nominal trajectory. Some typical applications are agricultural insecticide spraying operations and coastal surveying for which the vehicle has to follow a prespecified route [8]. These applications require the determination of waypoints and nominal trajectory, which is the task of mid-field and far-field mission planning. Standard optimal control techniques can be employed to obtain these trajectories. Some recent work in this direction, relevant to automated flight vehicles, have been reported in the literature [9–11].

We report here the systematic development of the basic structure and components of a radar-assisted collision avoidance/guidance strategy (RACAGS) that requires limited avionic aids and simple computations making it suitable for real-time applications. One of the major differences between the approach given here and those available in the literature is the emphasis we give on the *strategic* aspect of guidance and collision avoidance rather than the *systems* aspect that concerned most earlier papers. The other major difference is that we address the issues of both guidance and avoidance together as a combination inherent to the overall strategy of guiding a vehicle to a destination while avoiding enroute obstacles.

II. RADAR-ASSISTED COLLISION AVOIDANCE/GUIDANCE STRATEGY

The RACAGS uses a system of active sensors to provide range information about the obstacles in the environment ahead of the vehicle during low-altitude planar flight. The on-board guidance subsystem generates both guidance and avoidance commands based on the information received from the radar. The basic inputs to RACAGS are the guidance command input, the vehicle state estimate provided by the on-board inertial navigation system (INS), and the range map provided by the obstacle detection active sensors. The output is the commanded lateral acceleration.

A. Cone of Radar Returns

Consider Fig. 1 where a flight vehicle encounters a circular obstacle. The radar transmits signals in all directions in its field of view (FOV). A sufficiently wide field of view radar is assumed [12, 13]. The

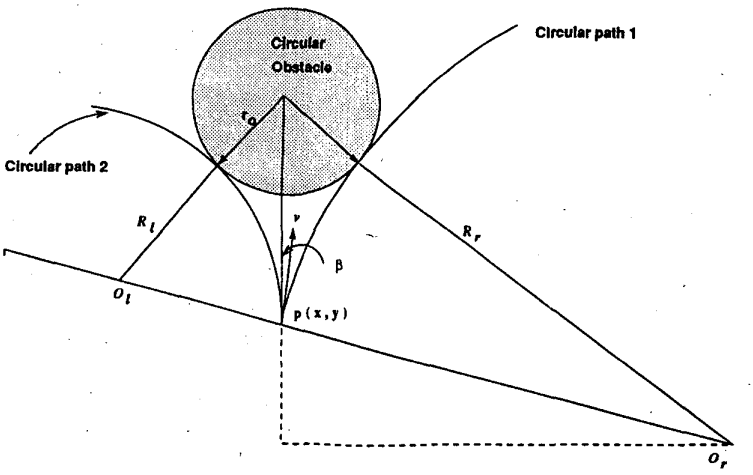


Fig. 2. Two possible collision-free paths.

on-board radar information is referenced with respect to the inertial coordinate system in an on-board inertial navigation unit. The reflected signals from the obstacle give information about the range to each point on the surface of the obstacle. In Fig. 1, the vehicle is represented by a point mass model with position coordinates (x,y). The two-dimensional geometry, formed with the points A and C, and the radar located at (x,y) is the *cone of radar signal returns*. The vehicle is assumed to move with a constant speed v. Here, r_0 is the radius of the obstacle, and R_0 is the line-of-sight (LOS) range to the center (x_0,y_0) of the obstacle. The coordinates of the destination or goal position is (x_g,y_g), and R_g is the LOS range to the goal. Obviously, if the velocity vector of the vehicle lies inside the *cone*, and continues to do so, then eventually there will be a collision with the obstacle. An avoidance command should enable the vehicle to execute a lateral maneuver to bypass the obstacle. If the velocity vector is outside the *cone*, the guidance command is sufficient to guide the vehicle toward the destination.

B. Guidance and Avoidance Commands

The guidance subsystem must generate two lateral acceleration commands—the *guidance* command and the *avoidance* command—to trace a collision-free trajectory from the start position to a goal position. The guidance command, denoted by a_{gd}, may be generated by some guidance strategy. For our purpose we use the classical proportional navigation (PN) guidance law [14]. From Fig. 1,

$$a_{gd} = C_{gd}\dot{\theta} \tag{1}$$

where C_{gd} is the guidance constant and

$$\dot{\theta} = -v\sin(\alpha-\theta)/R_g. \tag{2}$$

Thus, the computation of a_{gd} requires v, α, θ, and R_g. The current position of the flight vehicle v, and α are directly available from the inertial navigation unit, and θ and R_g can be computed easily since the goal point location is known. However, note that a different guidance strategy—for example, optimal control-based strategies—may also be used to generate the guidance command. The reason we use PN is that it requires minimal computations and the inputs needed are easily available from the radar and the INS carried in the vehicle.

The avoidance command is derived with reference to Fig. 2, where the position of the vehicle is shown as $p(x,y)$. To avoid collision, the vehicle can take a deviation to the right or left along either of the circular paths shown in Fig. 2. These are the circular paths with the largest possible radii (denoted by R_r and R_l) that would ensure collision avoidance. In order to ensure a collision-free trajectory, two important factors must be taken into account: 1) the latax (lateral acceleration) limit $a_{\max}$ of the vehicle, and 2) the location of the goal point.

The radii R_r and R_l can be obtained from Fig. 2 as,

$$R_r = \frac{R_0^2 - r_0^2}{2(r_0 - R_0\sin\beta)}, \qquad R_l = \frac{R_0^2 - r_0^2}{2(r_0 + R_0\sin\beta)} \tag{3}$$

where R_0 is the distance from the vehicle to the center of the obstacle as shown in Fig. 1. We must have $R_0 > r_0 > R_0|\sin\beta|$. The inequality $R_0 > r_0$ ensures that the vehicle is outside the obstacle and the inequality $r_0 > R_0|\sin\beta|$ implies that the velocity vector lies inside the cone of radar returns. If $r_0 < R_0|\sin\beta|$ then the velocity vector is outside the cone. When $r_0 = R_0|\sin\beta|$ the velocity vector lies along the boundary of the collision cone. In this case the vehicle will just graze the obstacle. The corresponding avoidance command (latax) that produces these circular paths

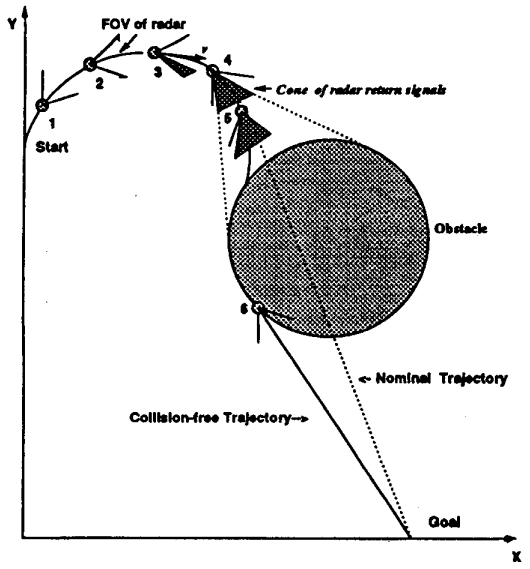


Fig. 3. Illustration of RACAGS.

can be obtained as

$$avc_r = -v^2/R_r, \qquad avc_l = v^2/R_l \tag{4}$$

where $|avc_r|$, and $|avc_l|$ are the minimum value of the magnitude of the avoidance latax commands which will divert the vehicle along the circular paths 1 or 2 (about the centers O_r and O_l), respectively, and ensure avoidance of the obstacle.

Although we have considered only a point mass model for the vehicle, its physical size can be taken into account by adding a suitable positive quantity to r_0 (the obstacle radius) while computing the avoidance commands. This is a standard technique in robotics literature [15] and has also been used in rotorcraft collision avoidance [1, 2].

C. Basic RACAGS Algorithm

The basic RACAGS algorithm is as follows.

If (Velocity vector not inside cone)
then $a_c = a_{gd}$
else if (Goal is on the right side of cone center line)
  then if $(a_{gd} \geq 0)$
    then $a_c = -\min\{a_{\max}, |avc_r|\}$
    else if $(|avc_r| \leq avc_l)$ *OR* $(|avc_r| \leq a_{\max})$
      then $a_c = \min\{a_{\max}, \max\{|avc_r|, |a_{gd}|\}\}$
      else $a_c = \min\{a_{\max}, avc_l\}$
  else if $(a_{gd} > 0)$
    then if $(|avc_r| > avc_l)$ *OR* $(avc_r \leq a_{\max})$
      then $a_c = \min\{a_{\max}, \max\{avc_l, a_{gd}\}\}$
      else $a_c = -\min\{a_{\max}, |avc_r|\}$
    else $a_c = \min\{a_{\max}, avc_l\}$

The basic concept behind RACAGS can be explained with reference to Fig. 3, where a radar with a 60° FOV is used. The nominal trajectory, generated using only the guidance command and ignoring the presence of the obstacle, passes through the obstacle. Several intermediate points on the trajectory of the vehicle are shown. The commands are assumed to be applied at certain intervals called the guidance cycle time (GCT). The points shown in the figure correspond to these intervals. At Points 1 and 2 it can be seen that the obstacle is not in the FOV of the radar. When the vehicle reaches Point 3, the obstacle is partially within the FOV of the radar. The shaded area shows the cone of radar signal returns. But the velocity vector of the vehicle is outside this cone and therefore the vehicle does not execute any avoidance maneuver. Because of this the vehicle continues to move along the nominal trajectory until Point 4. When the vehicle reaches Point 4, the obstacle is entirely within the FOV of the radar. Note that the transition from Point 3 to Point 4 appears to be discontinuous due to the non-zero GCT. The velocity vector is also inside the cone and so the vehicle is governed by the avoidance/guidance strategy. The avoidance command initiates lateral maneuvering of the vehicle so that the vehicle departs from the nominal trajectory. This process continues through Point 5 until the vehicle reaches Point 6 where the obstacle is no longer in the FOV of the radar. In this situation the guidance command is sufficient to guide the vehicle toward the goal.

III. CONSTRUCTION OF VIRTUAL OBSTACLES

In the preliminary development of the planar collision avoidance strategy, the radar returns from the obstacle are used to generate avoidance command to evade the obstacle. The computation of the avoidance command depended on the assumption that the obstacles are of circular shapes. This basic idea, with appropriate modifications, can be extended for obstacle avoidance in the real world environment. In the real world environment, natural obstacles such as mountains and trees have irregular shapes. Man-made obstacles such as buildings, towers, and bridges usually have well-defined, but not necessarily circular, geometrical shapes.

We propose an algorithm, called the *sliding circle algorithm*, for constructing the virtual obstacle. The logical criteria for an algorithm to construct a virtual obstacle to be acceptable should be the following. 1) The cone of radar returns from the original obstacle should be the same as the cone of radar returns from the virtual obstacle. 2) The virtual obstacle should be such that the part of the original obstacle facing the vehicle velocity vector must be wholly contained within the virtual obstacle.

The algorithm for construction of a virtual obstacle can be illustrated with reference to Fig. 4. First, we consider an arbitrary imaginary circle contained within

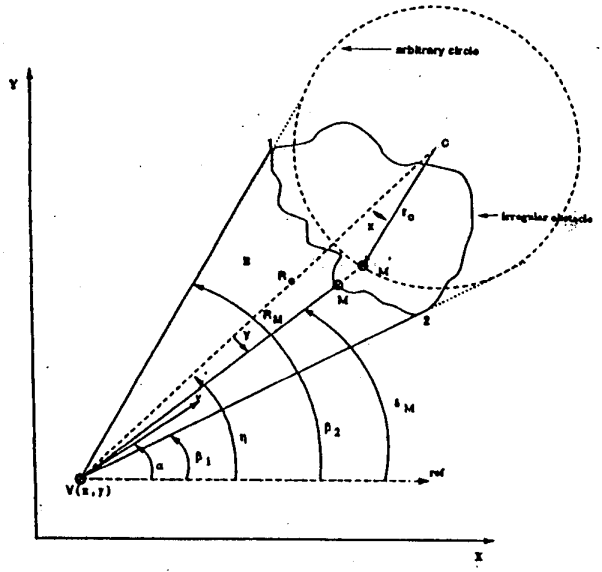


Fig. 4. Construction of virtual obstacle from arbitrary obstacle.

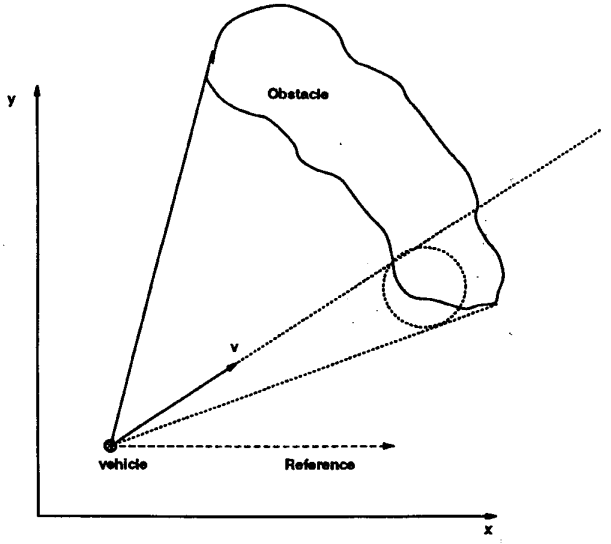


Fig. 5. Refinement of avoidance/guidance strategy.

the cone of radar returns from the actual obstacle and touching its boundaries. The center of the circle is assumed to be at an arbitrary distance R_0 from the vehicle. The radius of this circle is given by

$$r_0 = R_0 \sin((\beta_2 - \beta_1)/2). \tag{5}$$

Ideally, this circle should contain the part of the obstacle facing the vehicle and lying between the two edge points. Now, consider an arbitrary point M on the obstacle. Extension of the line OM will intercept the circle at point M'. Note that the range R_M is available from the radar returns at the angle δ_M from the reference. We can compute $R_{M'}$ as,

$$R_{M'} = R_0 \left(\cos\sigma_1 - \sqrt{\cos^2\sigma_1 - \cos^2\sigma} \right) \tag{6}$$

where

$$\sigma_1 = (\beta_1 + \beta_2)/2 - \delta_M, \qquad \sigma = (\beta_2 - \beta_1)/2. \tag{7}$$

To ensure that the point M on the obstacle lies inside the circle we must have,

$$R_{M'} < R_M \Rightarrow R_0 < R_M \Big/ \left(\cos\sigma_1 - \sqrt{\cos^2\sigma_1 - \cos^2\sigma} \right). \tag{8}$$

To ensure that all points on the portion of the obstacle facing the vehicle and lying between the two edge points should be contained in the circle, we must have,

$$R_0 = \min_{\delta_M \in [\beta_1, \beta_2]} R_M \Big/ \left(\cos\sigma_1 - \sqrt{\cos^2\sigma_1 - \cos^2\sigma} \right). \tag{9}$$

After obtaining R_0 from (9) we compute r_0 from (5) and then use RACAGS. The algorithm derives its name from the fact that we start with an imaginary circle within the cone of radar returns and then slide it outwards or inwards until it completely covers the part of the obstacle that faces the flight vehicle. Note that the virtual obstacle need not contain the original obstacle fully in the sense that on the farther side the original obstacle may project outside the imaginary circle. Implementation of this algorithm involves the selection of a few signal returns from inside the cone of radar returns. The larger this number of points is, better is the accuracy with which the virtual obstacle can be constructed to satisfy the two criteria. This number can be selected depending on the computation power available. In fact, quite often a single intermediate point (for example, the shortest distance point when this is distinct from the edge point) is good enough. When the shortest distance point is the same as one of the edge points then any intermediate point (for example, one which is on the centerline of the cone of radar returns) also serves the purpose reasonably well.

As the vehicle approaches the obstacle, the coordinates of the points used for creating the virtual obstacle will vary with time. So, at the beginning of each GCT, a new virtual obstacle is created and the basic collision avoidance strategy is used to detour around the obstacle.

It is possible to refine the sliding circle algorithm further to ensure that the imaginary circle from a large obstacle is constructed so as to cover only that part of the obstacle which poses a potential threat to the vehicle. For example, in Fig. 5, the virtual obstacle would be a large circle and the evasive maneuver prescribed by RACAGS may be to the right of the vehicle velocity vector. Because of the large size of the virtual obstacle the evasive maneuver level might be unnecessarily large. In which case, we can apply the sliding circle algorithm to only that part of the

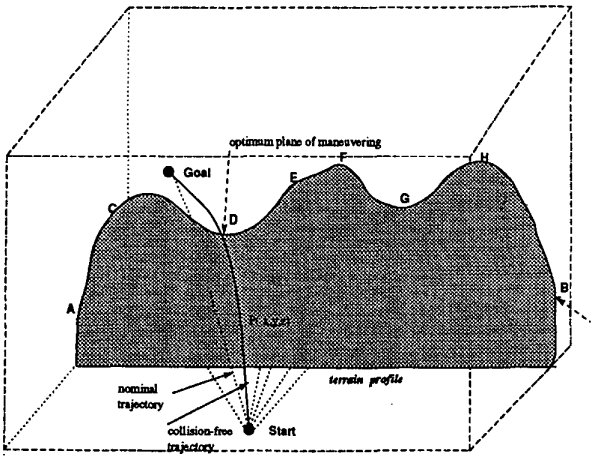


Fig. 6. Mountain range as obstacle.

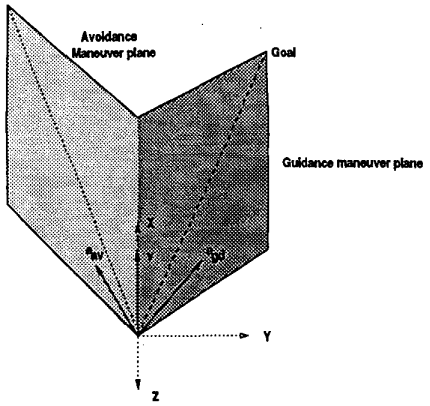


Fig. 7. Avoidance and guidance maneuver planes.

obstacle that lies on the right of the velocity vector and then recompute the evasive maneuver level. Of course, in this case, the avoidance/guidance strategy must also be modified to ensure that the vehicle takes an evasive maneuver only to the right.

IV. THREE-DIMENSIONAL COLLISION AVOIDANCE

In long-range missions the vehicle may encounter large hill ranges that stretch to great distances on both sides. In such scenarios the planar collision avoidance strategy applied on a horizontal plane will require the flight vehicle to take a large detour around the obstacle which may deplete its fuel unnecessarily. The only alternative is to fly above the obstacle while keeping in mind the covertness requirement [2]. All these scenarios give rise to applications of RACAGS to a three-dimensional flight scenario. This can be done by first determining a suitable plane for maneuvering and then using RACAGS on that plane. Suitability of a maneuver plane is determined by the requirements of how much concealment the plane allows.

A. Guidance and Avoidance Maneuver Plane

A guidance maneuver plane is defined as the plane containing the LOS vector from the current position of the vehicle to the goal point and the velocity vector of the vehicle. This is the plane in which the vehicle has to maneuver in order to reach the goal point. The guidance latax, denoted by a_{gd}, lies in this plane.

The choice of an avoidance maneuver plane must depend on a compromise between a fuel efficient trajectory and a covert trajectory. For example, in Fig. 6, lateral maneuvering in the NOE mode will require the vehicle to take too long a detour around the obstacle. It is more logical for the vehicle to fly over the hill range.

Suppose the vehicle follows a NOE mode of flight and selects a horizontal plane on which to maneuver. Consider the points A and B shown in Fig. 6. These points lie on a horizontal plane containing the current position of the vehicle and its velocity vector. The cone of radar returns on this horizontal plane would be fairly large as also the detour taken around the obstacle. Alternatively, the vehicle can select different vertical or slanting planes in space. In Fig. 6, the points C, D, E, F, G, and H are points on the top surface of the terrain that can be identified from the radar returns. Each of these points, along with the point P and the velocity vector of the flight vehicle can define an avoidance maneuver plane. Among these, the points D and G are valley points and are expected to provide maximum concealment. Suppose we select point D then the plane passing through D and P and containing the velocity vector of the vehicle becomes the avoidance maneuver plane.

B. Avoidance/Guidance Algorithm

After selecting the avoidance maneuver plane, the avoidance latax on this plane can be computed using the basic RACAGS algorithm. The difference between the planar application of RACAGS and the three-dimensional case is that in the present scenario the avoidance maneuver is only on one side, that is, the side that enables the vehicle to fly above the obstacle. So far as RACAGS is concerned, it is dealing with an obstacle that extends infinitely on one side and so the cone of radar returns on this side also extends till the extreme limits of the FOV of the radar.

The avoidance maneuver plane and the guidance maneuver plane both intersect at the velocity vector of the vehicle (see Fig. 7). The guidance command a_{gd} and the avoidance command a_{av} now need to be resolved as in the planar case. The resultant maneuver command will determine the motion of the vehicle. Note that as the vehicle moves in space, the guidance

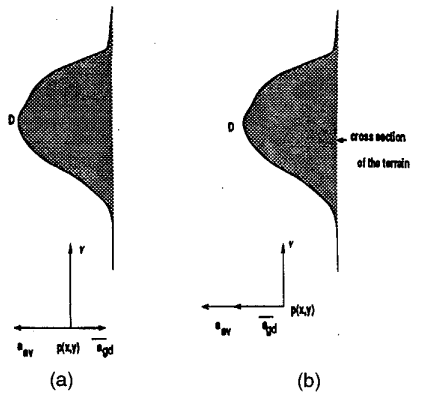


Fig. 8. Resolution of avoidance and guidance latex. (a) $a_t = \min\{a_{\max}, a_{av}\}$. (b) $a_t = \min\{a_{\max}, \max\{a_{av}, \bar{a}_{gd}\}\}$.

and avoidance maneuver planes, computed at the beginning of every GCT, also varies.

To resolve the guidance and avoidance commands we project the guidance latax a_{gd} on the avoidance maneuver plane. We denote this component as $\bar{a}_{gd}$. Fig. 8 shows the cross-section of the mountain cut by the avoidance maneuver plane (defined in Fig. 6 as the plane containing the vehicle velocity vector and the point D). The figure also shows the vehicle velocity vector, the avoidance latax a_{av}, and the component $\bar{a}_{gd}$ of the guidance latax. The actual latax, denoted by a_t, is obtained by the relations given in Fig. 8. In Fig. 8(a), the guidance latax is in the opposite direction to the avoidance latax and is ignored. In Fig. 8(b), the guidance latax is in the same direction as the avoidance latax and is taken into account in determining the actual maneuver command. This requires some minor and obvious modifications to the RACAGS algorithm.

V. WAYPOINT APPLICATIONS AND NOMINAL TRAJECTORY FOLLOWING

In principle, both these applications belong to the same class of problems with the difference that in the waypoint application the intermediate subgoals are at well separated locations on the trajectory whereas in the nominal trajectory following application the intermediate subgoals are spaced close together.

A. Waypoint Applications

Consider a flight vehicle on an NOE flight in a horizontal plane. The vehicle is required to pass through several prespecified intermediate goal points, spaced far apart, before flying to a specified final goal point. As the vehicle is required to fly at a low-altitude it has to avoid hills and other terrain on its way. It is assumed that the subgoals are well separated from each other and are not located inside an obstacle. This assumption is important in view

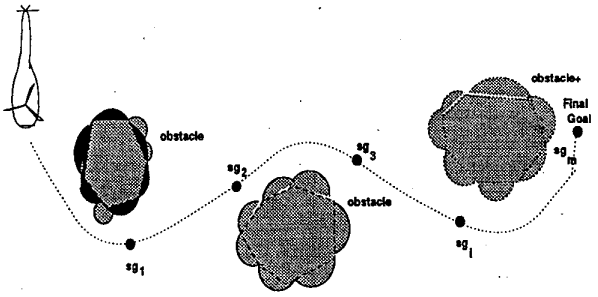


Fig. 9. Waypoint following in planar flight.

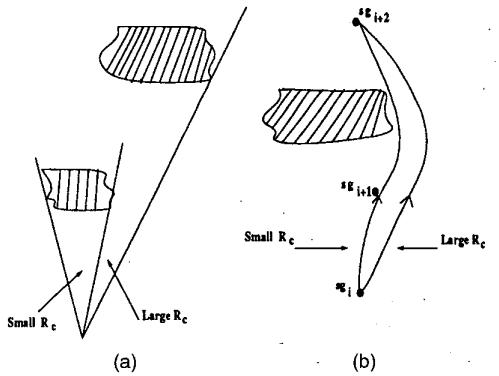


Fig. 10. Effect of R_c on trajectory.

of the fact that obstacles, by definition, are terrain and vegetation features that are not specified in the existing land mass database and thus could not be taken into consideration while fixing the waypoints. This assumption is somewhat strong and will be relaxed later in the paper when we consider the problem of nominal trajectory following where the nominal trajectory is specified by closely spaced waypoints. Fig. 9 shows a waypoint following application in a NOE mode of flight.

Intermediate goal points (waypoints or subgoals) are denoted by $sg_1, sg_2, \ldots, sg_m$ and their coordinates are stored in the form of a list of sequential goals in the guidance computer. Here, sg_1 is the first subgoal and sg_m is the final goal. As the vehicle starts on its mission, it considers sg_1 as its first goal and generates the lateral maneuver commands according to the RACAGS algorithm. When sg_1 has been achieved the current goal point is updated to sg_2. This process continues until the vehicle reaches its final goal point sg_m. This is a straightforward algorithm for selecting and updating the current goal point. Nevertheless, the following few important issues need to be addressed.

Radar Cut-Off Range R_c: The selection of R_c is crucial here. As the vehicle has to pass through specified goal points, a large R_c can make the vehicle deviate too much from the desired trajectory because of radar returns from distant obstacles that can make the cone of radar returns unnecessarily large (see Fig. 10(a)). Also, the vehicle may be prevented to

reach a subgoal lying between an obstacle and the vehicle, by avoidance maneuver generated by the collision avoidance strategy (see Fig. 10(b)). Thus, the value of R_c has to be small while at the same time it should be large enough to warn the vehicle well in advance of an impending collision. The actual selection of R_c has to depend on some knowledge of the distribution of the obstacles on the terrain.

Determination of the Current Goal Point: The goal point needs to be updated as the vehicle achieves the current goal point. Since we have assumed that the goal points are always outside the obstacles on the terrain, this is a straightforward process. When a subgoal in the list of subgoals is achieved, the next subgoal in the list automatically gets the status of the current goal point.

Achievment of Goal Point: There should be an effective mechanism to determine whether a goal point has been achieved or not. It is not realistic to insist on the condition that the vehicle must pass exactly through a subgoal. Hence, we assume that a goal point has been achieved either 1) when the vehicle comes within a certain acceptable range R_g from the goal point, or 2) when the closest approach to the current goal point has occurred. The second condition is necessary for those cases where the vehicle fails to come within the acceptable range R_g to the goal point due to the close vicinity of an obstacle.

B. Nominal Trajectory Following

Many mission scenarios need precise tracking of the nominal trajectory. An example of such a nominal trajectory is shown in Fig. 11. There could be several obstacles on the nominal trajectory that prevent the vehicle from following a NOE flight on the horizontal plane. The objective of a guidance/avoidance strategy that avoids the obstacles on its path would be to ensure as much of the nominal trajectory following as is possible without risk of collision with the obstacles.

The algorithm we propose here is an extension of the waypoint algorithm proposed in the previous section. In the nominal trajectory following algorithm we define a large number of goal points in close proximity to each other on the nominal trajectory. However, the large number of subgoals gives rise to the problem of handling subgoals that fall inside the obstacles or that are in close proximity to it. Obviously, these subgoals are either unachievable or have a high risk of collision. The basic problem here is to design an efficient algorithm to define the current subgoal for the flight vehicle. In the literature the selection of the current goal point has been recognized as an important problem [2], where it is assumed to be a reference point that moves with the vehicle at some fixed length ahead of it. This method has a drawback that if the nominal trajectory passes through a large

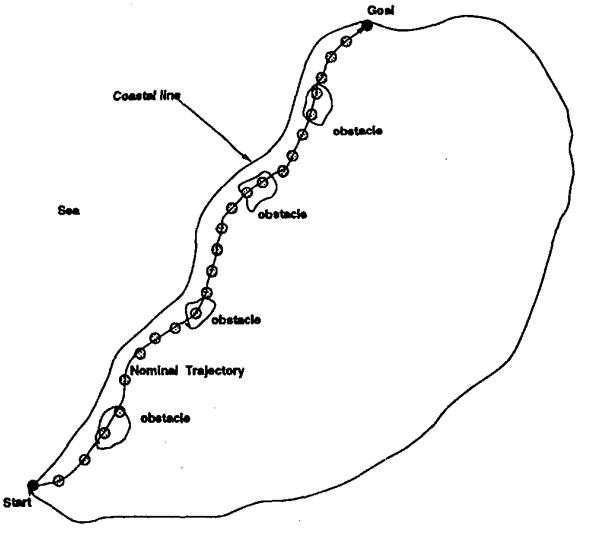


Fig. 11. Subgoals for nominal trajectory following.

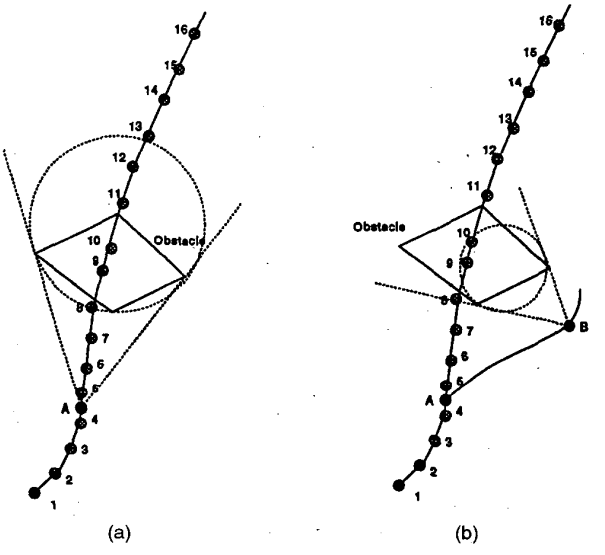


Fig. 12. Creation of goal lists.

obstacle the reference point may remain inside it for a long duration and thus increase the risk of collision with the obstacle. In the algorithm given below, we adopt a different approach that avoids this risk. The essential idea behind this algorithm is to create several lists of goal points and then manipulate them to obtain the current goal point to be used by the guidance system to generate the guidance command. These lists are created at the beginning of a GCT and are defined as follows. (See Fig. 12 where the obstacle has a rhombus-like shape and the subgoals are marked with small shaded circles on the nominal trajectory.)

Current_Goal_List: This contains the list of goal points that the vehicle has to pass through in future if it follows the nominal trajectory. For example, in Fig. 12(a), when the vehicle is at point A on the

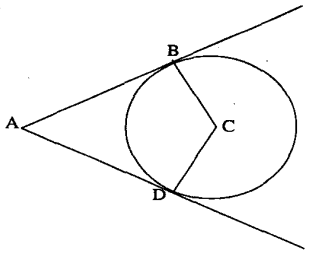


Fig. 13. Convex regions for determining Obstacle_Goal_List.

nominal trajectory,

$$\text{Current\_Goal\_List} = \{sg_5, sg_6, \ldots, sg_m\}.$$

When the vehicle is not on the nominal trajectory then the Current_Goal_List is created by updating the previous Current_Goal_List. We see how it is done in the Subgoal_Selection algorithm given below.

Obstacle_Goal_List: This contains the list of all subgoals that fall within the cone and the circle created by the sliding circle algorithm. Thus, in Fig. 12(a), when the vehicle is at point A,

$$\text{Obstacle\_Goal\_List} = \{sg_5, sg_6, \ldots, sg_{12}\}.$$

When the vehicle is at point B, we have,

$$\text{Obstacle\_Goal\_List} = \{sg_9, sg_{10}\}.$$

The determination of the set of goal points that constitutes the Obstacle_Goal_List is a straightforward operation in which the cone and the circle are represented as two bounded convex regions, one polyhedral and the other circular (see Fig. 13). Note that after the circular virtual obstacle has been defined by the RACAGS algorithm through the position of its center C and its radius, the two convex regions are the polyhedron ABCD and the imaginary circle defining the virtual obstacle. It is easy to determine which of the goal-points lie in either of these regions.

Unattainable_Goal_List: This contains the list of all subgoals that lie inside the cone but in the region between the vehicle and the obstacle. Thus, in Fig. 12(a), when the vehicle is at point A,

$$\text{Unattainable\_Goal\_List} = \{sg_5, sg_6, \ldots, sg_8\}.$$

When the vehicle is at point B, the Unattainable_Goal_List is empty. Note that the creation of the unattainable goal list needs the radar return signals from the directions in which the goal points lie. The essential idea is to select those goal points in the obstacle goal list as unattainable goals that have distances from the radar smaller than the obstacle distance in that direction.

Temporary_Current_Goal_List: Obtained by subtracting the Unattainable_Goal_List and the Obstacle_Goal_List from the Current_Goal_List. The first element in the Temporary_Current_Goal_List is the current goal point of the vehicle.

Below we present an algorithm for determining the current subgoal to which the vehicle guides during a guidance cycle.

ALGORITHM Subogoal_Selection
At the beginning of a GCT,
Step 1 If (Any goal has been achieved in the previous GCT period)
then
Current_Goal_List = Current_Goal_List—{Achieved goals and all subgoals prior to it}.
Step 2 If (An obstacle is within distance R_c from the vehicle)
then
Determine the Obstacle_Goal_List.
Determine the Unattainable_Goal_List.
Current_Goal_List =
Current_Goal_List—Unattainable_Goal_List.
Temporary_Current_Goal_List =
Current_Goal_List—Obstacle_Goal_List.
Current_Subgoal = First element in the Temporary_Current_Goal_List
else
Current_Subgoal = First element in the Current_Goal_List.

In Fig. 12, this algorithm yields sg_{13} as the current subgoal when the vehicle is at point A and sg_{11} as the current subgoal when the vehicle is at point B.

Another important point to note is that the contents of the obstacle goal list depends on the value of the radar cut-off range R_c which defines the virtual obstacle and the cone. A large R_c will create a large obstacle goal list and consequently a large number of subgoals will remain unattainable. Too small an R_c will increase the risk of collision with the obstacle. Also, the lists are created even when the velocity vector is not inside the cone of radar returns. This is necessary to ensure that any achievable subgoal is not omitted. For example, in Fig. 12(b), when the vehicle is at point B, the velocity vector is outside the cone of radar returns, but it is necessary to create the temporary goal list in order to have sg_{11} as the current goal point.

VI. SIMULATION RESULTS

In this section we present a few simulation results to illustrate the efficacy of RACAGS when it is applied to a real world environment with obstacles of arbitrary shapes. The basic vehicle and guidance data is as given in Table I. Note that these values, as also the dimensions of the obstacles, have been scaled up from realistic values in order to be able to illustrate the finer points of RACAGS. The algorithm works equally well with the scaled down realistic values.

A. Planar Obstacle Avoidance

In Fig. 14 the flight vehicle encounters an irregularly shaped obstacle. The virtual obstacle is

TABLE I
Basic Vehicle and Guidance Data

Vehicle velocity v	300 m/s
C_{gd}	1500
Latax limit $a_{\max}$	20 g

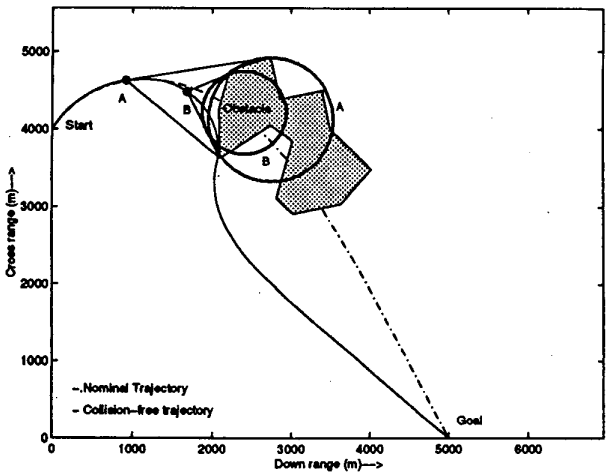


Fig. 14. Collision avoidance using sliding circle algorithm.

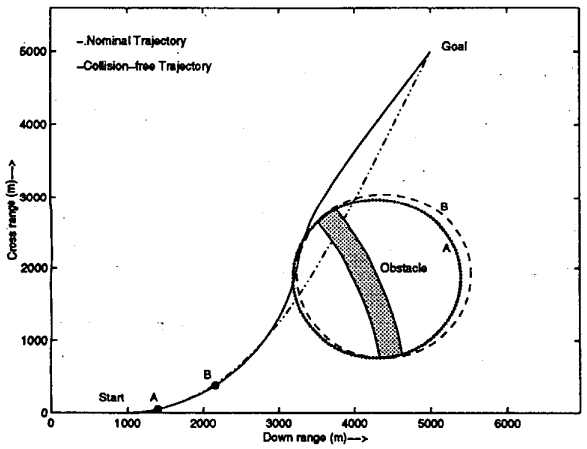


Fig. 15. Collision avoidance with concave obstacle.

constructed at the beginning of each GCT (assumed to be 2.5 s) using the sliding circle algorithm.

Fig. 15 shows a concave obstacle that the vehicle may often encounter in the real world environment. The GCT is 0.5 s. Two circles are shown at points A and B on the trajectory using the sliding circle algorithm. The nominal trajectory here passes very close to one of the edges of the obstacle and consequently the avoidance maneuver needed is low.

To show the effect of changing the GCT, the collision-free trajectory using RACAGS is shown with a concave obstacle in Fig. 16. The trajectories for

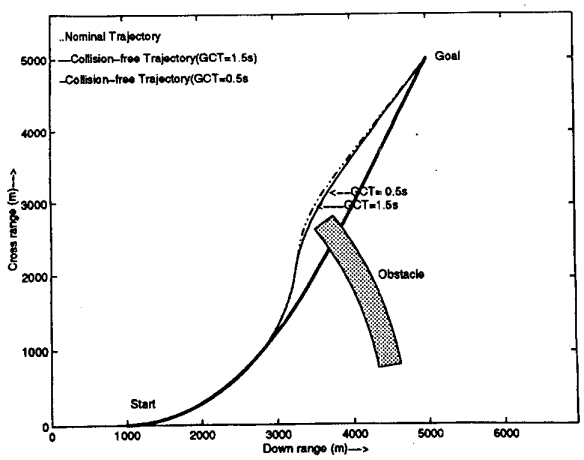


Fig. 16. Influence of guidance cycle time.

two different GCTs (0.5 s and 1.5 s) are shown. Note that the gap between the obstacle and the avoidance trajectory increases with increase in GCT. A zero GCT would have enabled the vehicle to graze the obstacle.

The collision avoidance strategy developed here is equally applicable in the real world environment with more than one obstacle inside the FOV of the radar. If all the obstacles are clustered close together, the signal returns will appear to form a single cone of radar returns and the sliding circle algorithm will create a single virtual obstacle. On the other hand, if the obstacles are well separated then multiple virtual obstacles will be created.

In Fig. 17(a) we have three irregularly shaped obstacles. The radar cut-off range is 0.9 km. Because of this short cut-off range, the radar sees only one obstacle at a time and consequently passes through them. But this poses a potential threat to the vehicle since there is a distinct possibility that it may find its path blocked after entering between the first two obstacles. In the next simulation (Fig. 17(b)) we show the same obstacle configuration as in Fig. 17(a) but with the radar range cut-off as 2 km. Because of the large value of R_c the obstacles appear as a single obstacle to the radar and the flight vehicle takes a large detour around the cluster of obstacles.

Obstacle avoidance and collision-free trajectory planning is also an important area of research in robotics [15]. An extensive comparison with robotic path planning methods is beyond the scope of this work. But we illustrate how the basic principles of RACAGS can be used to guide robots or automated guided vehicles through a collision-free path. We assume that these robots are equipped with radars or laser scanners. In a recent paper, a local navigation technique with obstacle avoidance has been proposed for mobile robots [16]. In this method, the only information needed about the local environment is the distance between the robot and the obstacles in

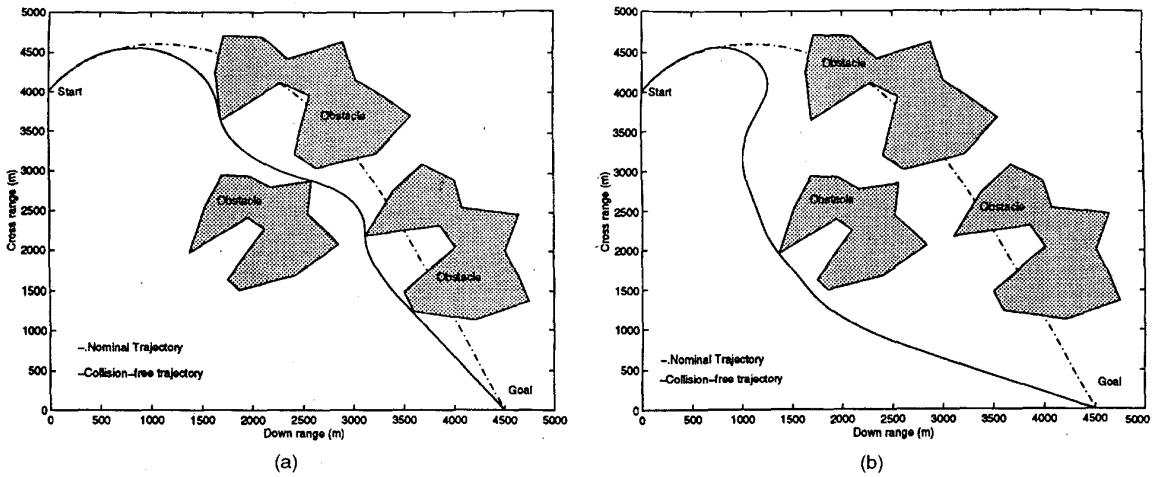


Fig. 17. Collision avoidance with (a) $R_c = 0.9$ km. (b) $R_c = 2$ km.

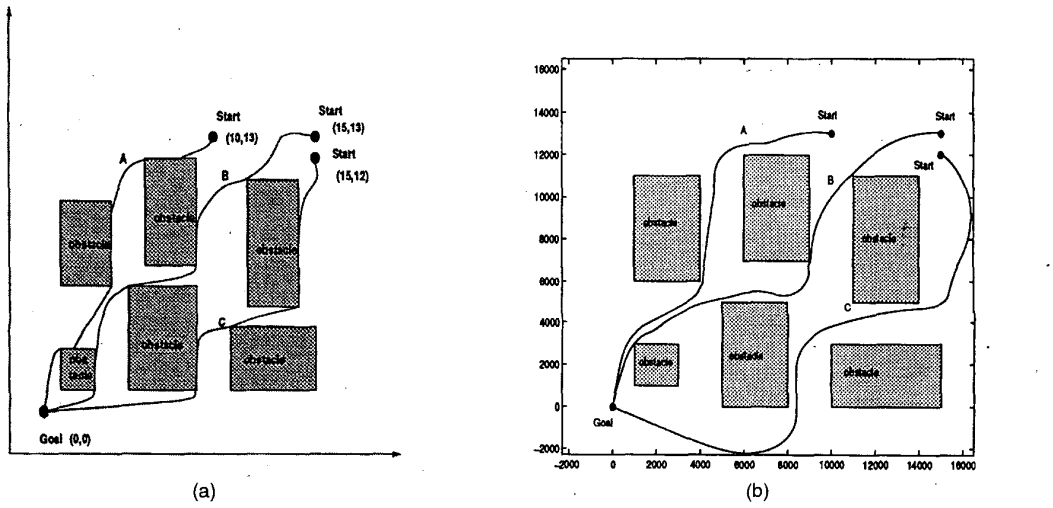


Fig. 18. (a) Adaptive navigation technique for robots. (b) Collision avoidance for RACAGS.

three specified directions. The method gives rise to robot trajectories as shown in Fig. 18(a). The robot starts from three different starting points with different initial path angles. Collision-free paths are obtained in all the cases.

Fig. 18(b) shows the same environment. RACAGS is applied to obtain collision-free paths in all three cases. The basic data is the same as in Table I. The radar cut-off range is set at $R_c = 2.5$ kms and the GCT is 0.5 s. We see that RACAGS is quite effective in comparison to the adaptive navigation technique. A point to note is that in the adaptive navigation technique the trajectories hug the obstacle surfaces whereas the RACAGS trajectories keep some clearance from the obstacles. The reason for this difference is that RACAGS tries to avoid a virtual obstacle rather than the original obstacle. Also, it maintains the forward speed of the vehicle constant while limiting the lateral acceleration that can be applied. This puts a lower limit on the turn radius of the vehicle.

B. Three-Dimensional Collision Avoidance

In Fig. 19, a profile of the terrain is shown. For the simulation we assume a mountain range with a flat face. The start and the goal positions of the vehicle are as shown. The basic simulation data is given in Table II. The radar return signals from the terrain presents a three-dimensional range-map. We assume a sufficiently wide FOV radar. Several inclined planes are selected for maneuvering. Among these planes the one which provides the best concealment—in the sense that the vehicle requires to attain the minimum altitude to fly over the terrain obstacle—is selected as the optimum plane of maneuver. From Fig. 19, it

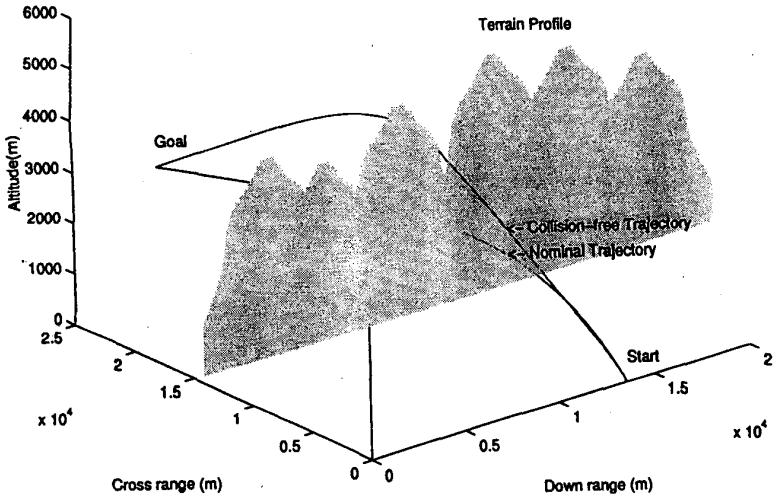


Fig. 19. Terrain avoidance with best covertness and concealment.

TABLE II
Basic Data for Simulation

Vehicle velocity	300 m/s
Heading angle α	90°
Flight path angle γ	15°
Starting Point	(13568,0) m
Goal Point	(13568,40000) m
C_{gd}	1500
Guidance cycle time	2.5 s
Latax limit a_{max}	20 g

is obvious that the plane of maneuver shown in the figure is the best plane for covert penetration.

C. Obstacle Avoidance in Waypoint Applications

The basic flight vehicle data is as given in Table I. The GCT is 0.5 s, R_g is 0.5 km, and R_c is 3.6 kms. The nominal trajectory is obtained by ignoring the obstacles and considering the goal points successively to generate the guidance commands according to the PN guidance law. In the simulation study shown in Fig. 20 the vehicle encounters nine obstacles in its path and it has three subgoals. Two different values of R_c = 20 kms and 5 kms are used. Until the fourth obstacle the trajectories for both values of R_c are more or less the same. The fifth obstacle is detected well in advance by the system with the longer cut-off range (R_c = 20 kms) making the vehicle take a deviation to the left. The system with the low range cut-off (R_c = 5 kms) detects the obstacle much later when it has already advanced to the right side of the obstacle and therefore takes a deviation to the right. This simulation illustrates the effect of varying the range cut-off on the collision-free path.

In the next simulation study we consider a real world environment where the vehicle has to avoid obstacles of arbitrary shapes and sizes. A typical

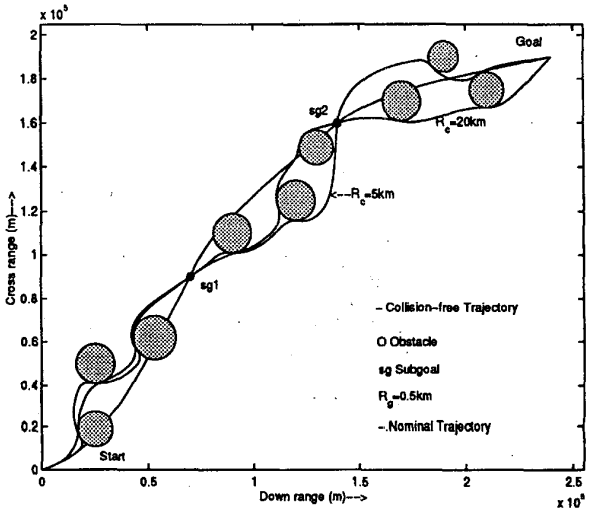


Fig. 20. Following intermediate waypoints through multiple obstacles with two different radar range cut-offs.

simulation result is obtained in the real world environment. Fig. 21 shows two obstacles in the nominal path of the vehicle. There are three goal points. The vehicle successfully passes through the waypoints while avoiding obstacles in the path. The values R_c = 1 km and R_g = 0.5 km are assumed.

D. Obstacle Avoidance in Nominal Trajectory Following

The basic data is given in Table I. The radar cut-off range R_c is 3 kms and R_g is 500 m. In Fig. 22 we have two circular obstacles with the nominal trajectory defined by a large number of intermediate goal points (shown as small black circles). The trajectory followed by the vehicle using the algorithm given here is shown in the figure. Initially, when

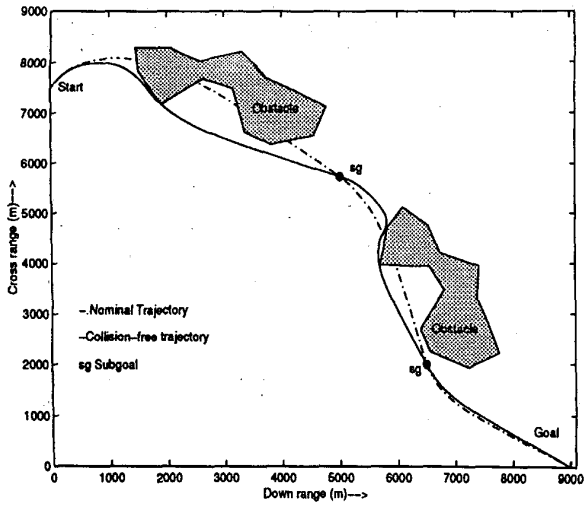


Fig. 21. Following intermediate waypoints in real world environment.

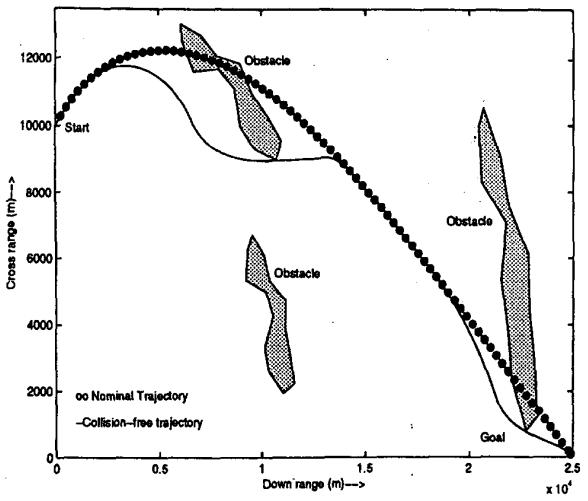


Fig. 23. Nominal trajectory following through two hills.

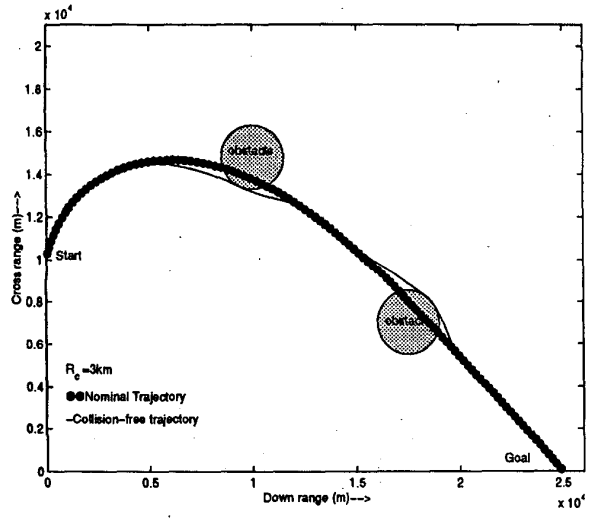


Fig. 22. Nominal trajectory following through two circular obstacles.

the first obstacle is still not within the range R_c, the Current_Goal_List contained all the subgoals ahead of the vehicle. The list gets updated whenever a subgoal is achieved. As soon as the obstacle appears within the range R_c the various lists are created and the current goal point shifts to a point beyond the obstacle. The avoidance/guidance strategy then prescribes a lateral maneuver that takes the vehicle around the obstacle until its trajectory joins the nominal trajectory. A similar process takes place when the second obstacle is encountered.

In Fig. 23 we have a real world environment with the nominal trajectory passing through two arbitrarily shaped obstacles. Here too, R_c is 3 kms. A similar process takes place here and the vehicle successfully evades the obstacle. It can be seen that the vehicle takes an evasive maneuver well in advance and so misses quite a few subgoals. This can be rectified by using a small value of R_c or by using the refinement of the sliding circle algorithm discussed earlier.

VII. CONCLUDING REMARKS

This paper addresses the task of obstacle avoidance and guidance as a combined operation from the strategic point of view. The RACAGS algorithm is developed to handle irregularly shaped obstacles in a planar environment. The basic technique is extended to three-dimensional collision avoidance and finally to waypont applications and nominal trajectory following. The main objective behind the algorithm development was minimal computational effort. Future work in this direction would address the implementation issues that integrate actual sensor and guidance system with the proposed algorithm.

REFERENCES

[1] Bhanu, B., Das, S., Roberts, B., and Duncan, D. (1996)
A system for obstacle detection during rotorcraft low altitude flight.
IEEE Transactions on Aerospace and Electronic Systems, **32**, 3 (July 1996), 875–897.

[2] Cheng, V. H. L. (1990)
Concept development of automatic guidance for rotorcraft obstacle avoidance.
IEEE Transactions on Robotics and Automation, **6**, 2 (Apr. 1990), 252–257.

[3] Cheng, V. H. L., and Sridhar, B. (1991)
Considerations for automated nap-of-the earth rotorcraft flight.
Journal of the American Helicopter Society, **36**, 2 (Apr. 1991), 61–69.

[4] Lin, C. F. (1991)
Modern Navigation, Guidance, and Control Processing.
Englewood Cliffs, NJ: Prentice-Hall, 1991.

[5] Kaminer, I., Pascoal, A., Hallberg, E., and Silvestre, C. (1998)
Trajectory tracking for autonomous vehicles: An integrated approach to guidance and control.
Journal of Guidance, Control, and Dynamics, **21**, 1 (Jan.–Feb. 1998), 29–38.

[6] Davis, W. R., Kosicki, B. B., Boroson, D. M., and Kostishack, D. F. (1996)
Micro air vehicles for optical surveillance.
Lincoln Laboratory Journal, **9**, 2 (1996), 197–214.

[7] Fleury, P. A. (1986)
Covert penetration systems: Future strategic aircraft missions will require a new sensor system approach.
In *Proceedings of the IEEE National Aerospace and Electronics Conference* (May 1986), 220–226.

[8] Costello, D., Kaminer, I., Carder, K., and Howard, K. (1995)
The use of unmanned vehicle systems for coastal ocean surveys: Scenarios for joint underwater and air vehicle missions.
In *Proceedings of the Workshop on Intelligent Control of Autonomous Vehicles*, Lisbon, Portugal, 1995, 61–72.

[9] Hess, R. A., and Jung, Y. C. (1989)
An application of generalized predictive control to rotorcraft terrain-following flight.
IEEE Transactions on Systems, Man, and Cybernetics, **19**, 5 (Sept.–Oct. 1989), 955–962.

[10] Shapira, I., and Ben-Asher, J. Z. (1997)
Near-optimal horizontal trajectories for autonomous air vehicles.
Journal of Guidance, Control, and Dynamics, **20**, 4 (July–Aug. 1997), 735–741.

[11] Kaminer, I., Pascoal, A., Hallberg, E., and Silvestre, C. (1998)
Trajectory tracking for autonomous vehicles: An integrated approach to guidance and control.
Journal of Guidance, Control, and Dynamics, **21**, 1 (Jan.–Feb. 1998), 29–38.

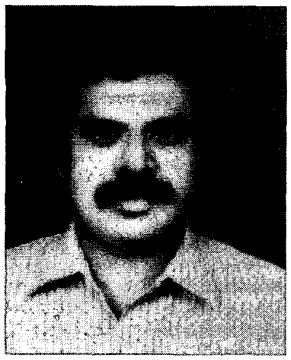
[12] Cheng, V. H. L., and Lam, T. (1992)
Automatic guidance and control laws for helicopter obstacle avoidance.
In *Proceedings of the IEEE International Conference on Robotics and Automation*, Nice, France, May 1992, 252–260.

[13] Cerchie, P., Shipley, B., Aust, R., Wasson, J., and Reago, D. (1992)
Manned simulation results concerning design parameters for an effective obstacle avoidance system (OASYS).
Journal of the American Helicopter Society, **37**, 2 (Apr. 1992), 3–10.

[14] Zarchan, P. (1994)
Tactical and Strategic Missile Guidance (2nd ed.).
Washington, DC: AIAA, 1994.

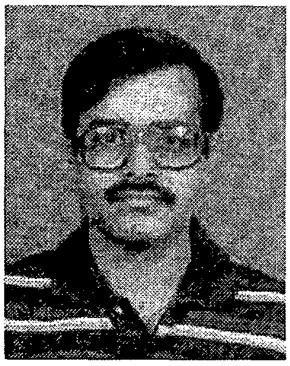
[15] Fujimura, K. (1991)
Motion Planning in Dynamic Environments.
New York: Springer-Verlag, 1991.

[16] Fujimori, A., Nikiforuk, P. N., and Gupta, M. M. (1997)
Adaptive navigation of mobile robots with obstacle avoidance.
IEEE Transactions on Robotics and Automation, **13**, 4 (Aug. 1997), 596–602.



B. Ajith Kumar obtained his B.Sc. (Engg) degree in electrical engineering from the University of Kerala, India, in 1984 and the M.Tech. degree from the Indian Institute of Technology, Madras, in 1993. He obtained his Ph.D. degree in engineering from the Indian Institute of Science, Bangalore, in 1999.

At present he is an Assistant Professor in the Department of Electrical Engineering at the College of Engineering, Trivandrum, India. His areas of research interest includes guidance and control of aerospace vehicles and robot path planning problems.



Debasish Ghose obtained a B.Sc. (Engg) degree from the Regional Engineering College, Rourkela, India in 1982, and an M.E. and a Ph.D. degree, from the Indian Institute of Science, Bangalore, in 1984 and 1990, respectively.

He is presently an Associate Professor of Aerospace Engineering at the Indian Institute of Science. He has held short and long-term visiting positions at several other universities, the most recent of which was in the Mechanical and Aerospace Engineering Department of the University of California at Los Angeles, where he spent a sabbatical year during 1999–2000. He was also selected for the Alexander von Humboldt fellowship in the year 2000. His areas of research interest are in applications of game theory, guidance and control of aerospace and robotic vehicles, and scheduling problems in distributed computing systems.