Rendezvous-Guidance Trajectory Planning for Robotic Dynamic Obstacle Avoidance and Interception

Faraz Kunwar and Beno Benhabib

***Abstract*—This correspondence presents a novel online trajectory-planning method for the autonomous robotic interception of moving targets in the presence of dynamic obstacles, i.e., position and velocity matching (also referred to as rendezvous). The proposed time-optimal interception method is a hybrid algorithm that augments a novel rendezvous-guidance (RG) technique with the velocity-obstacle approach, for obstacle avoidance, first reported by Fiorini and Shiller. The obstacle-avoidance algorithm itself could not be used in its original form and had to be modified to ensure that the online planned path deviates minimally from the one generated by the RG algorithm. Extensive simulation and experimental analyses, some of which are reported in this correspondence, have clearly demonstrated the tangible time efficiency of the proposed interception method.**

***Index Terms*—Online trajectory planning, rendezvous guidance (RG), target interception.**

I. Introduction

Robotic environments have, typically, time-varying topologies, with several objects moving with respect to each other. A common thread to such environments is that the topology of the environment is often uncontrollable and, therefore, requires online planning and execution of the robotic-vehicle trajectories, i.e., a need for autonomous routing decisions. For example, current automated guided vehicles' (AGVs) autonomy, as well as their efficiency, could be tangibly increased, if they were capable of making online routing decisions to avoid mobile or static obstacles [1]. In this context, the focus of this correspondence is on the two autonomy aspects of the robotic vehicles: 1) guidance-based time-optimal rendezvous with a moving target (matching position and velocity) and 2) obstacle avoidance.

The problem of robotic-vehicle interception in the obstacle-cluttered environments, using a rendezvous-guidance (RG) method augmented with a modified exact cell decomposition (MECD) method for the time-optimal rendezvous, was first addressed in [2]. This method, although computationally efficient, has several shortcomings, including lack of time optimality. In order to overcome the shortcomings of the proposed original method and achieve an improved rendezvous with the dynamic targets, this correspondence augments the RG method with the velocity-obstacle (VO) approach [3].

A. Guidance-Based Interception

Missile-guidance techniques have been classified into five main categories [4], [5]: line-of-sight (LOS) guidance; pure pursuit (PP); proportional navigation guidance (PNG); optimal guidance (OG); and other guidance methods, including the use of the differential game theory. The missile-guidance laws assume that the future trajectory of the target is completely defined by either an analytical or a probabilistic model [6]–[8]. However, the problem of the velocity matching introduced in this correspondence has not been an issue in missile-guidance applications.

The PNG law uses the homing triangle for computing the acceleration of an interceptor pursuing an evading target. The homing triangle is defined by the interceptor, the target, and the point of interception. This control law makes the interceptor's acceleration normal to its path and proportional to the rate of change of the LOS vector to the target. Due to its low computational requirements, the simplicity of an onboard implementation, and the time-optimality characteristics, the PNG has been the most widely used guidance technique [9].

The need for velocity matching has resulted in a new class of guidance methods, commonly referred to as RG methods. A PNG-based RG method for the docking problem of two space vehicles was proposed in [10]. In [11], the use of an exponential-type guidance was suggested for asteroid rendezvous. The problem of the rendezvous, with an object capable of performing evasive maneuvers in order to avoid the rendezvous, was addressed in [12].

The utilization of a guidance-based technique in robot motion planning, with the purpose of improving upon the interception time achievable by visual-servoing techniques, was first reported in [13]–[16]. Although these works showed that guidance-based methods could yield shorter interception times compared to other available techniques, all were limited to environments with no obstacles. This correspondence proposes a new guidance-based method that overcomes this limitation by enabling it to optimally deal with the obstacles.

B. Obstacle Avoidance

Motion-planning problems for mobile robots have been classified as static or dynamic. In the former, all of the obstacle information is known to the planner prior to planning. In the latter case, the information about the environment becomes known to the planner only during runtime and often during the execution of a partially constructed plan. The static methods for obstacle avoidance, like the potential-field (PF) techniques, roadmaps, and CD methods [17]–[20], calculate the desired motion direction and steering commands in two separate steps. In the first step, the obstacle-avoidance method provides the intermediate destination points that connect a collision-free path from the robot to the target. In the second step, the acceleration commands are derived for the path generated for the motion of the robot. Such a methodology would not be acceptable for a dynamic environment with fast moving obstacles, where the uncertainty about the environment prevents the computation of a solution that is guaranteed to succeed.

The curvature-velocity (CV) [21] and the dynamic window (DW) [22] methods are based on the steer-angle-field approach [23]. The CV method chooses a location in the translational- and rotational-velocity space that satisfies the constraints placed on the robot and maximizes an objective function [24]. The lane-curvature method (LCM) [25] improves upon the CV method by using a directional-lane method. The DW method considers the kinematic and dynamic constraints of a mobile robot [26]. The kinematic constraints are taken into account by directly searching the velocity space of the robot. The search space is reduced to a DW representing the velocities achievable by the robot in a given interval of time. In spite of the good results for obstacle avoidance at high velocities achieved by both the CV and DW methods, the local-minima problem persists. In order to overcome this shortcoming, the DW method was integrated with a gross-motion planner in [27] and extended to use a map in conjunction with the sensory information in [28] to generate the collision-free motions.

Manuscript received May 6, 2005; revised November 21, 2005 and March 16, 2006. This work was supported by the Natural Sciences and Engineering Research Council of Canada. This paper was recommended by Associate Editor F. Hoffmann.

The authors are with the Computer Integrated Manufacturing Laboratory, Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, ON M5S 3G8, Canada (e-mail: kfaraz@mie.utoronto.ca; beno@mie.utoronto.ca).

Digital Object Identifier 10.1109/TSMCB.2006.877792

However, these approaches require *a priori* information about the environment.

The algorithm proposed in this correspondence uses the VO approach first proposed in [3]. This method, like the CV and DW methods, considers the kinematic and dynamic constraints of the robot in computing the acceleration commands. However, the VO method determines the potential collisions and computes the collision-free paths for the robots moving in the dynamic environments. The VO represents the robot velocities that would cause a collision with an obstacle in some future time. An avoidance maneuver is computed by selecting the velocities that are outside the VO. In order to ensure that the maneuver is feasible, the dynamic constraints are mapped into the velocity space. The VO method was extended in [29] to include objects moving along the nonlinear trajectories. A comparison of the DW and the extended VO approaches, for obstacle avoidance, was reported in [30], which showed that the VO approach clearly performs more favorably than would the DW approach.

II. Proposed Hybrid Interception Method

A schematic diagram of the proposed implementation of the hybrid RG–VO method is shown in Fig. 1. A vision module first obtains the positional and velocity $(\boldsymbol{p}, \boldsymbol{v})$ state vectors of all the objects in the workspace, namely, the robot's $(\boldsymbol{p}_R, \boldsymbol{v}_R)$, the target's $(\boldsymbol{p}_{\mathrm{T}}, \boldsymbol{v}_{\mathrm{T}})$, and the obstacles' $(\boldsymbol{p}_{O1}, \boldsymbol{v}_{O1})$, $(\boldsymbol{p}_{O2}, \boldsymbol{v}_{O2})$, etc., and passes this information to the RG and VO algorithms. Based on this information, the proposed hybrid path planner evaluates whether an obstacle is within the time horizon t_{h}, i.e., whether the time-to-impact between an obstacle and the target is less than a threshold period. If no obstacle is within the time horizon, then, the position and velocity information are passed on to the RG method, which in turn selects the maximum possible closing velocity, $\boldsymbol{v}_{\mathrm{RG}}$, required to rendezvous with the target for the next sampling instant Δt. Finally, a lateral acceleration command $\boldsymbol{a}_{\mathrm{RG}}$ to achieve the desired velocity in the interval $(t_i + \Delta t)$ is computed and passed on to the robot. If, on the other hand, it is determined that an obstacle is within the time horizon t_{h}, the positional and velocity information are passed on to the VO method. The VO method first constructs a VO cone for each obstacle in the time horizon t_{h}. This cone represents a set of relative colliding velocities between the robot and the obstacle. By selecting a velocity $\boldsymbol{v}_{\mathrm{VO}}$ that is not within the VO set, the obstacle avoidance can be ensured for the next sampling period. Finally, a lateral acceleration command $\boldsymbol{a}_{\mathrm{VO}}$ to achieve the desired velocity in the interval $(t_i + \Delta t)$ is computed.

A. Interception via RG

Let us define a LOS as the relative position vector $\boldsymbol{r}$ connecting the interceptor/robot to the target, as shown in Fig. 2. The parallel-navigation rule states that the relative velocity $\dot{\boldsymbol{r}}$ between the robot and the target should remain parallel to $\boldsymbol{r}$ at all times [4]. If this rule holds throughout the motion of the interceptor, the distance between the interceptor and the target would decrease until they collide. Furthermore, if the target moves with a constant velocity, the parallel navigation results in a global time-optimal interception.

The parallel-navigation law is expressed by the following two relationships:

$$\boldsymbol{r} \times \dot{\boldsymbol{r}} = 0 \tag{1}$$

$$\boldsymbol{r} \cdot \dot{\boldsymbol{r}} < 0. \tag{2}$$

Equation (1) guarantees that the LOS and the relative velocity remain parallel, while (2) ensures that the interceptor is not receding from the target. The above equations can be solved for $\dot{\boldsymbol{r}}$ in a parametric form to yield

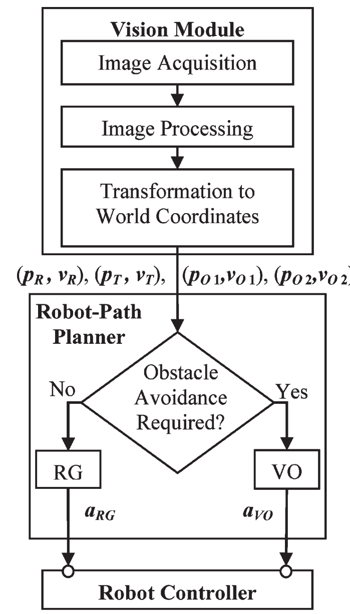


Fig. 1. Schematic of the proposed RG–VO method.

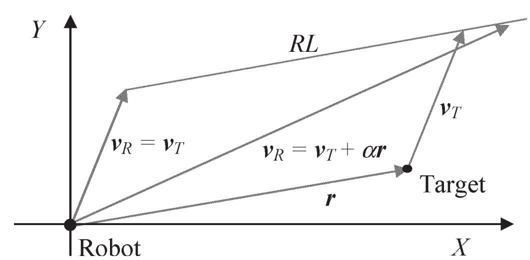


Fig. 2. Construction of RL.

$$\dot{\boldsymbol{r}} = -\alpha\, \boldsymbol{r} \tag{3}$$

where α is a positive real number. The instantaneous relative velocity, also referred to as the "closing velocity," can then be written in terms of the robot and target velocities, denoted by $\boldsymbol{v}_R$ and $\boldsymbol{v}_{\mathrm{T}}$, respectively, as follows:

$$\dot{\boldsymbol{r}} = \boldsymbol{v}_{\mathrm{T}} - \boldsymbol{v}_R. \tag{4}$$

Substituting (3) into (4) and solving for the robot velocity yields

$$\boldsymbol{v}_R = \boldsymbol{v}_{\mathrm{T}} + \alpha\, \boldsymbol{r}. \tag{5}$$

The vectors $\boldsymbol{r}$ and $\boldsymbol{v}_{\mathrm{T}}$ are determined using the data received from a vision module, based on the instantaneous positions of the robot and the target. Substituting these two known vectors into (5) would result in a locus for the robot's velocity vectors $\boldsymbol{v}_R$, all lying on a semiline parameterized by α. This semiline, referred to as the rendezvous line (RL), is depicted in Fig. 2. The center of the coordinate frame is located on the robot to show the instantaneous relative position of the target. The endpoints of the velocity vectors show the position of the target or the robot after one sampling period, should they adopt the corresponding velocities. If the robot continually adopts a velocity command that falls on the instantaneous RL, the direction of the LOS would remain constant, and the positional matching between the robot and the maneuvering target is guaranteed.

In order to rendezvous with a target, the velocity of the robot/interceptor must also match the velocity of the maneuvering target at the time of the interception. The velocity commands generated based on (5) guarantee the position matching. Thus, the next task is to find an α value such that velocity matching is also assured.

Let us assume that, from the current instant until interception, the robot is guided by the velocity commands that lie on the instantaneous RL. This assumption allows us to consider the interception problem only in the direction of the LOS. Let us, furthermore, consider that the acceleration capability of the robot in this direction is given by A. This acceleration would be used to bring the closing velocity down to zero. Assuming a constant acceleration for the rest of the robot motion, the simultaneous reduction of the velocity and position differences in the direction of the LOS for interception may then be written as

$$\begin{cases} \dot{\boldsymbol{r}}_{\max}^{\text{rend}} - At_r = 0 \\ \boldsymbol{r} - \dot{\boldsymbol{r}}_{\max}^{\text{rend}} t_r + 1/2\, At_r^2 = 0 \end{cases} \tag{6}$$

where $\dot{\boldsymbol{r}}_{\max}^{\text{rend}}$ is the magnitude of the maximum allowable closing/rendezvous velocity (hence, the superscript rend) and t_r is the time remaining to intercept the target from the current instant. The maximum instantaneous allowable closing velocity is then obtained by solving (6)

$$\dot{\boldsymbol{r}}_{\max}^{\text{rend}} = \sqrt{2\boldsymbol{r}A}. \tag{7}$$

The maximum closing velocity, as imposed by the frequency of the velocity-command generation by the trajectory planner for a fast asymptotic interception, is given by

$$\dot{\boldsymbol{r}}_{\max}^{\text{cr}} = \boldsymbol{r}/n\Delta t. \tag{8}$$

The value of n is determined experimentally. The final allowable closing-velocity component of the velocity command is then obtained by considering (7) and (8) simultaneously

$$\boldsymbol{v}_{\max}^{\text{rel}} = \min \left\langle \dot{\boldsymbol{r}}_{\max}^{\text{rend}}, \dot{\boldsymbol{r}}_{\max}^{\text{cr}} \right\rangle. \tag{9}$$

The end points of all the velocity-command vectors on the RL that have a closing-velocity component smaller than $\boldsymbol{v}_{\max}^{\text{rel}}$ constitute a line segment extending from $\boldsymbol{v}_R = \boldsymbol{v}_{\text{T}}$ to $\boldsymbol{v}_R = \boldsymbol{v}_{R,\max} (= \boldsymbol{v}_{\text{T}} + \boldsymbol{v}_{\max}^{\text{rel}}(\boldsymbol{r}/\|\boldsymbol{r}\|))$. This set of points is referred to herein as the rendezvous set (RS) [Fig. 3(a)]. The velocity represented by $\boldsymbol{v}_{\max}^{\text{rel}}$ [Fig. 3(a)] may not be achievable by the robot within the sampling period Δt. Therefore, we define a feasible velocity region (FVR) representing all the velocities achievable by the robot within Δt, taking into account the kinematic and dynamic constraints on the robot [3]. This region is depicted by the polygon in Fig. 3(b). The velocity selected by the robot for the sampling interval Δt is the component of the RS within the FVR with the maximum value, which is represented by $\boldsymbol{v}_R(t_i + \Delta t)$ in Fig. 3(b). It is, thus, concluded that, if the robot adopts the velocity commands from within the RS with the largest allowable closing-velocity component, then, a time-efficient interception can be achieved.

B. Obstacle Avoidance via VO Method

Herein, for simplicity, we model the robot and the obstacles as circles, thus, considering a planar problem with no rotations. This is a reasonable assumption since the general polygons can be represented by a number of circles.

1) Planar VO: Let us consider two circular objects, a robot (R) and an obstacle (O), at time t_i, with velocities $\boldsymbol{v}_R$ and $\boldsymbol{v}_O$, respectively [Fig. 4(a)]. In order to compute the VO, we first map the obstacle model into the configuration space of the robot by reducing the robot model to a point $\hat{R}$ and enlarging the obstacle by the radius of the robot model to $\hat{O}$. The state of the moving object is represented by its position and a velocity vector attached to its center. Next, the set of the colliding relative velocities between $\hat{R}$ and $\hat{O}$, called the collision cone, CC_{RO}, is defined as

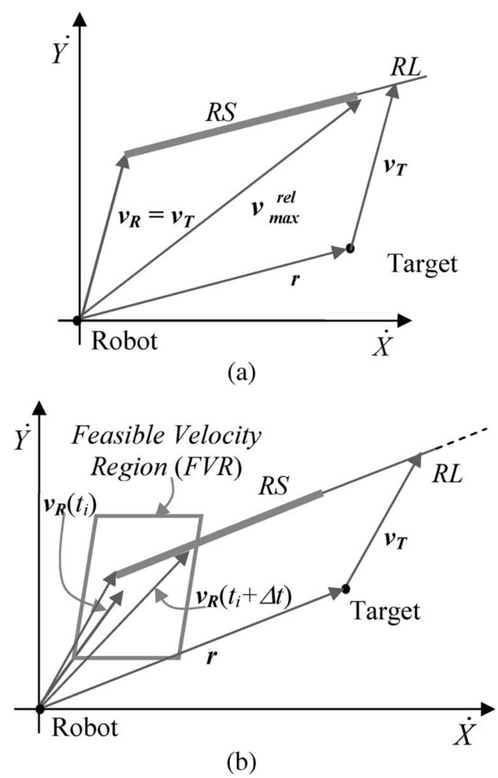


Fig. 3. Generation of the rendezvous command.

$$\text{CC}_{RO} = \{v_{R,O} | \lambda_{R,O} \cap \hat{O} \neq 0\} \tag{10}$$

where v_{RO} is the relative velocity of $\hat{R}$ with respect to $\hat{O}$, $v_{R,O} = \tilde{v_R} - v_O$, and $\lambda_{R,O}$ is the line along the direction of v_{RO}. This cone is the light gray sector with the apex in $\hat{R}$, bounded by the two tangents λ_f and λ_r from $\hat{R}$ to $\hat{O}$ [Fig. 4(b)]. Any relative velocity that lies between the two tangents to $\hat{O}$, λ_f and λ_r, will cause a collision between R and O. Therefore, any relative velocity outside CC_{RO} is guaranteed to be collision-free, provided that the obstacle $\hat{O}$ maintains its current speed. Furthermore, selecting a velocity in front of λ_f will cause a front-avoidance maneuver, allowing the robot to pass in front of the obstacle, whereas selecting a velocity behind λ_r would cause a rear-avoidance maneuver, allowing the robot to pass behind the obstacle.

Each collision cone is specific to a particular pair of robot and obstacle. To consider the multiple obstacles, we must establish an equivalent condition to the absolute velocities of R. This is done simply by adding the velocity of O $\boldsymbol{v}_O$ to each velocity in CC_{RO} and forming the VO

$$\text{VO}_O = \text{CC}_{RO} \oplus v_O \tag{11}$$

where $\oplus$ is the Minkowski-vector sum operator, as shown in Fig. 4(b) by the dark gray sector. The VO partitions the absolute velocities of R into the avoiding and colliding velocities. Selecting $\boldsymbol{v}_R$ outside of VO would avoid collision with O. Velocities on the boundaries of VO would result in R grazing O or

$$R(t_i) \cap O(t_i) = 0, \qquad \text{if } v_R(t_i) \notin \text{VO}_O(t_i). \tag{12}$$

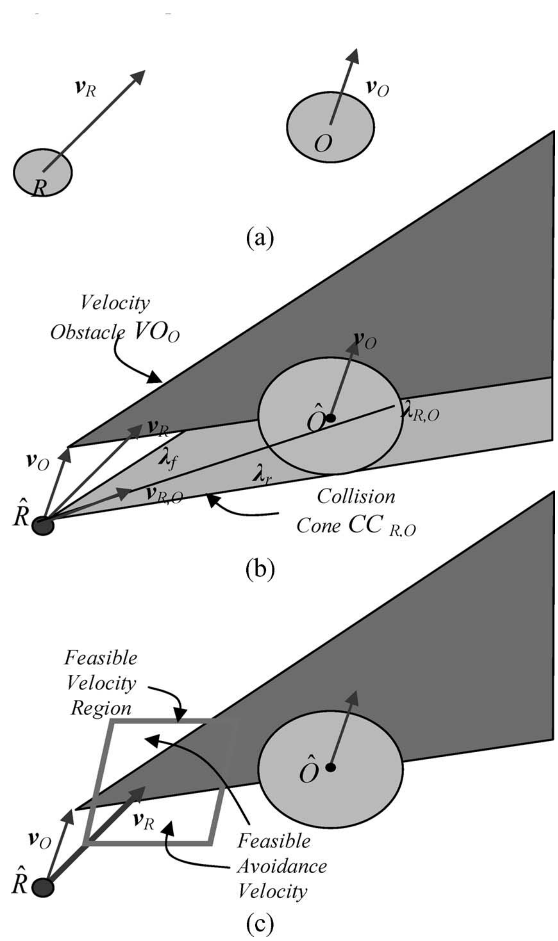


Fig. 4. VO method for obstacle avoidance. (a) Two colliding objects. (b) VO_O. (c) FAV.

In order to avoid multiple obstacles, we consider the union of the individual VOs of (10)

$$\mathrm{VO} = \cup_{i=1}^{m} \mathrm{VO}_{O_i} \tag{13}$$

where m is the number of obstacles. The avoidance velocities, then, consist of those velocities $\boldsymbol{v}_R$ that are outside all the VOs.

2) Avoidance Maneuver: An avoidance maneuver consists of a one-step change in the velocity to avoid a future collision within a given time horizon. The new velocity must be achievable by the robot. Thus, the set of the avoidance velocities are also limited by the FVR defined above. It is represented by the polygon shown in Fig. 4(c).

The set of feasible velocities $\mathrm{FV}(t_i + \Delta t)$ over the sampling period Δt is, thus, defined as

$$\mathrm{FV}(t_i + \Delta t) = \{v | v = v_R(t_i) \oplus \Delta t \bullet \mathrm{FA}(t_i)\} \tag{14}$$

where $\mathrm{FA}(t_i)$ represents the set of feasible accelerations of the robot at time t_i. This defines the FVR, as shown by the polygon in Fig. 4(c). It is computed by scaling $\mathrm{FA}(t_i)$ by Δt and adding it to the current velocity of the robot $\boldsymbol{v}_R$, as shown schematically in Fig. 5.

The set of feasible avoidance velocities (FAV) is defined as the difference between the feasible velocities and the VO

$$\mathrm{FAV}(t_i + \Delta t) = \mathrm{FV}(t_i + \Delta t) \Theta \mathrm{VO}(t_i) \tag{15}$$

where Θ denotes the operation of the set difference. A maneuver avoiding obstacle O can then be computed by selecting any velocity in the FAV. Fig. 4(c) shows schematically the set FAV, consisting of

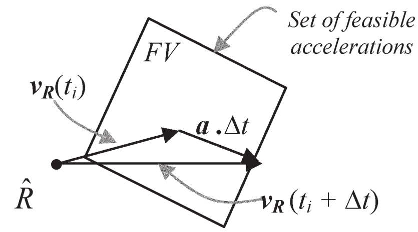


Fig. 5. Feasible accelerations.

two disjoint closed subsets. For the multiple obstacles, the FAV may consist of the multiple disjoint subsets. It is also possible to choose the type of avoidance maneuver by selecting which side of the obstacle the mobile robot will pass. The most feasible velocity for the next sampling instant is selected by performing a heuristic search.

3) Heuristic Search Strategies: Heuristics are designed to satisfy a prioritized series of goals. In our case, avoiding obstacles is the primary goal of the robot, and reaching the desired target by minimizing a performance index and selecting a desired trajectory structure are the secondary goals. The hierarchical fulfillment of these goals is intrinsic to the VO approach, since choosing the velocities within the FAV at each instant guarantees that all obstacles are avoided, provided that a safe avoidance velocity exists within the FAV. Then, depending on the appropriate selection of avoidance velocity, some of the secondary goals can also be satisfied.

It is evident that not all velocities within the FAV are candidates for the avoidance maneuvers, since they may move the robot away from its target, or they may produce a very slow trajectory. Furthermore, other considerations may affect the choice of avoidance velocity, such as the type of obstacle that is to be avoided, the speed of the obstacle, the size of the obstacle, etc. These considerations can be used to approximately classify the elements in the environment into two broad categories: high-risk and low-risk obstacles. The natural heuristic strategy used in the presence of high-risk obstacles is to let them pass without crossing their path. This heuristic maps into a rear-avoidance maneuver. The avoidance of a low-risk obstacle can be achieved with a more risky maneuver such as a front-avoidance maneuver. It is important to note that there is no guarantee that any objective is achievable at any time. The purpose of the heuristic search is to find an acceptable local solution if one exists.

Based from above, the following three basic heuristics were defined in [4].

1) Select the highest safe velocity along the line to the goal [Fig. 6(a)] so that the trajectory takes the robot toward its target. This strategy is denoted by to goal (TG).
2) Select the maximum safe velocity within some specified angle θ, from the line to the goal [Fig. 6(b)], so that the robot moves fast even if this implies not aiming directly at the goal. This second strategy is called the maximum velocity (MV).
3) Select the velocity that avoids the obstacles according to their perceived risk (high or low) [Fig. 6(c)]. Here, the obstacle is perceived to be of high risk; therefore, the chosen velocity is selected to be the largest among the rear-avoidance velocities. This strategy is called the structure (ST).

III. Implementation

The procedure in generating the desired velocity $\boldsymbol{v}_R\ (t_i + \Delta t)$ for the next sampling interval, in which the acceleration command is based, is discussed herein. Fig. 7 represents a general case, where a robot is intercepting a target in the presence of two moving obstacles. Furthermore, it is assumed that both obstacles are within t_h; thus,

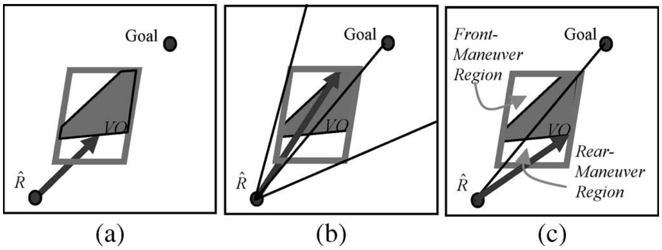


Fig. 6. Heuristic strategies. (a) TG. (b) MV. (c) ST.

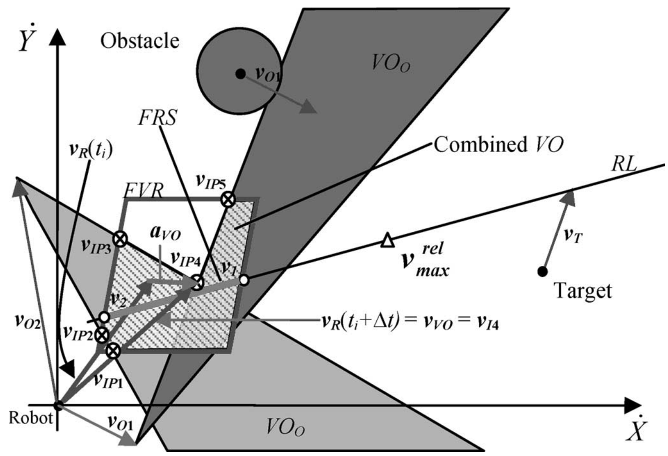


Fig. 7. Trajectory planning by combining the RG with the VOs.

avoidance of both obstacles is necessary. Here, $\boldsymbol{v}_R\,(t_i)$ represents the velocity of the robot at time instant t_i; $\boldsymbol{v}_{O1}$ is the velocity of the first obstacle; and $\boldsymbol{v}_{O2}$ is the velocity of the second obstacle at the same time instant.

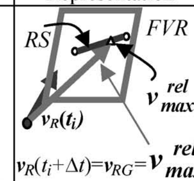
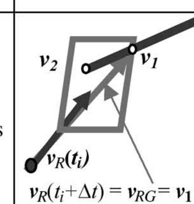
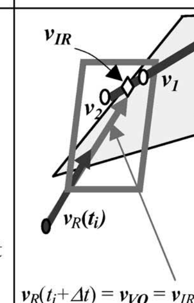
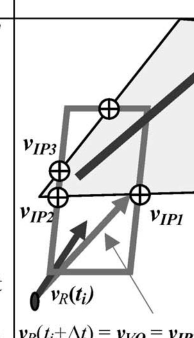
First, the algorithm constructs a combined VO for all the obstacles within t_h; this is represented by the shaded region in Fig. 7. This is done simply by subtracting the areas of the VOs of both the obstacles, VO_{O1} and VO_{O2}, which lie within the FVR from the area of the FVR. The FV set of the robot for the next sampling interval $\boldsymbol{v}_R\,(t_i+\Delta t)$, to avoid the obstacle and rendezvous with the target in any situation, is given by (16). At any instant, only one velocity is selected from this set to avoid all the obstacles at that particular instant. The velocities represented in (16) are also shown in Fig. 7, where

$$\boldsymbol{v}_R(t_i+\Delta t) = \left\{\boldsymbol{v} | \boldsymbol{v} \in \left[\boldsymbol{v}_{\max}^{\mathrm{rel}}, \boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_{\mathrm{IR}}, \boldsymbol{v}_{\mathrm{IP1-IP5}}\right]\right\}. \quad (16)$$

In (16), $\boldsymbol{v}_{\max}^{\mathrm{rel}}$ represents the maximum allowable closing-velocity component, computed by the RG algorithm required for rendezvous; $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ represent the velocities followed by the robot based purely on the RG algorithm without any obstacle-avoidance requirements; and $\boldsymbol{v}_{\mathrm{IR}}$ represents the intersection point of the VO with the RL. It is to be noted that this point only exists if a portion of the RL is outside the VO: it is the MV within the FVR, which satisfies the RG and VO methods when obstacle avoidance is necessary. The points $\boldsymbol{v}_{\mathrm{IP1}}-\boldsymbol{v}_{\mathrm{IP5}}$ represent the intersection points of the combined VO with the FVR. These represent the maximum and minimum front- and rear-avoidance velocities, respectively, which may be selected to ensure that the robot avoids the obstacle. The number of these intersection points $(1-n)$ at any instant depends on the number and orientation of the obstacles present in the workspace and also within the t_h. For a single obstacle, the maximum number of the intersection points is four.

Out of all the velocities given by (16) and detailed above, only one is selected for the execution in the next sampling period $(t_i+\Delta t)$. In order to determine which velocity is selected, a sequence of checks is performed (Table I).

TABLE I
GENERATION OF THE ROBOT VELOCITY COMMAND

Seq	Check	Action	Schematic Representation
1	No obstacle is within the time horizon, t_h, and $v_{\max}^{rel}$ is within *FVR*.	In this case, the maximum closing velocity component, $v_{\max}^{rel}$, is within *FVR*. The desired velocity $\boldsymbol{v}_R(t_i+\Delta t)$ for the next time interval is set to $v_{\max}^{rel}$ as shown. This velocity is obtained from the RG method.	
2	No obstacle is within the time horizon, t_h, and $v_{\max}^{rel}$ is not within *FVR*.	The algorithm determines both $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. Out of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, the desired velocity $\boldsymbol{v}_R(t_i+\Delta t)$ for the next time interval is the one which takes the robot nearest to the target. In this example, this velocity is $\boldsymbol{v}_1$. This velocity is obtained from the RG method.	
3	An obstacle is within the time horizon, t_h, and $v_{\max}^{rel}$ is within the *VO*. Furthermore, a portion of the *RS* within the *FVR* is outside the *VO*.	The intersection point of the edge of the *VO* cone that intersects with the *RS* is determined by the algorithm and denoted by $\boldsymbol{v}_{IR}$. The proximity of this point to the target is compared with $\boldsymbol{v}_1$ or $\boldsymbol{v}_2$ (whichever is outside the cone. In this case, $\boldsymbol{v}_2$). The velocity which takes the robot closer to the target, $\boldsymbol{v}_{IR}$, is selected as the desired velocity $\boldsymbol{v}_R(t_i+\Delta t)$ for the next time interval. In this case the desired velocity is obtained from the VO method.	
4	An obstacle is within the time horizon, t_h, and the entire *RS* and $v_{\max}^{rel}$ are inside the *VO*.	Any point selected in the *FRS* will lead to collision with the obstacle. In order to get the desired velocity, the intersection of the *VO* cone with the *FVR* is obtained: this corresponds to four points in the *FVR* for a single obstacle, $\boldsymbol{v}_{IP1}$ to $\boldsymbol{v}_{IP4}$. These points are examined to obtain the velocity which takes the robot closest to the target, in this case, $\boldsymbol{v}_{IP1}$, which is selected as the desired velocity $\boldsymbol{v}_R(t_i+\Delta t)$ for the next time interval. This velocity is obtained from the VO method.	

Based on the velocity obtained from Table I, the algorithm computes the final acceleration command that is sent to the robot controller, as expressed below

$$\boldsymbol{a}_R(t_i+\Delta t) = \begin{cases} v_{\mathrm{RG}} - v_R(t_i)/\Delta t & \text{obstacle avoidance not required} \\ v_{VO} - v_R(t_i)/\Delta t & \text{obstacle avoidance required.} \end{cases} \quad (17)$$

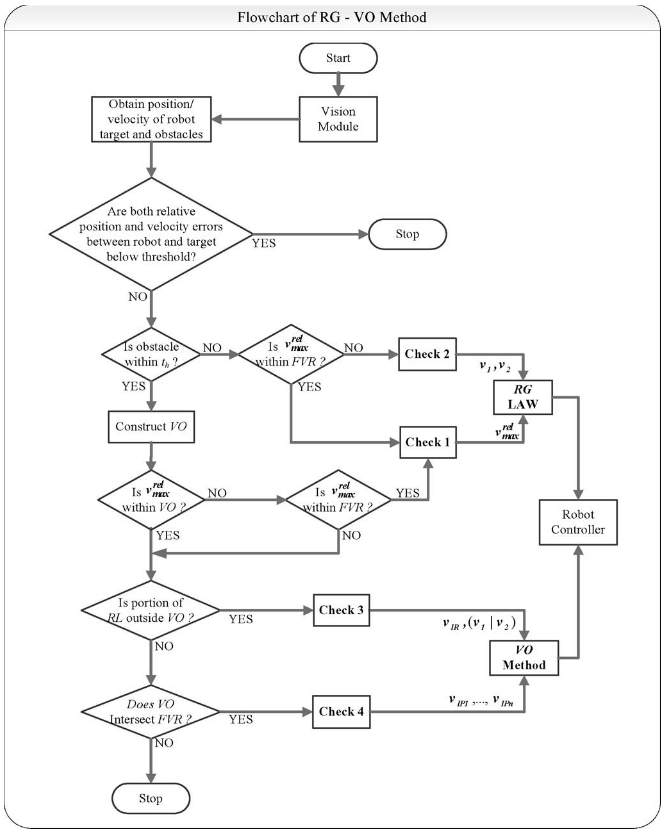


Fig. 8. Flowchart of the RG–VO algorithm.

TABLE II
INTERCEPTION DATA SIMULATIONS

S/NO	Complexity	Obs Vel (mm/s)		Tgt Vel (mm/s)	Interception Time (s)	
		Obs. 1	Obs. 2		MECD	VO
1	Moving Obs - Moving Tgt	85	0	100	5.78	5.2
2	Moving Obs - Moving Tgt	90	60	80	11.7	8.3

A flowchart detailing the steps carried out by the robot to the rendezvous with the target is given in Fig. 8.

IV. SIMULATIONS

A number of simulations were carried out for the rendezvous using the proposed hybrid RG–VO algorithm. A comparison of this method was also carried out with the RG–MECD method presented in [2]. The MV and the lateral acceleration of the robot were limited to 300 mm/s and 3000 mm/s^2, respectively, in all the examples. The robot is assumed to have no axial acceleration and is initially moving at 300 mm/s in the positive X direction. The criterion for a successful interception was set as $<$ 10-mm relative distance in both the X and Y directions and a relative velocity of $<$ 10 mm/s.

The results of two of the simulations comparing the RG–MECD method with RG–VO method are given in Table II and Fig. 9. Fig. 9(a) shows a simulation carried out with one obstacle in motion and with a maneuvering target, whereas Fig. 9(b) shows a simulation in which both the obstacles as well as the target are moving. A simulation showing a complex scenario with a large number of moving objects is shown in Fig. 9(c), using only the RG–VO method. A time sequence of the objects in the environment, to show the time evolution of the avoidance maneuver, is shown in Fig. 9(d). Fig. 9(d) depicts the same scenario, as shown in Fig. 9(c), but, in this case, the position of each object at the discrete time instants is shown.

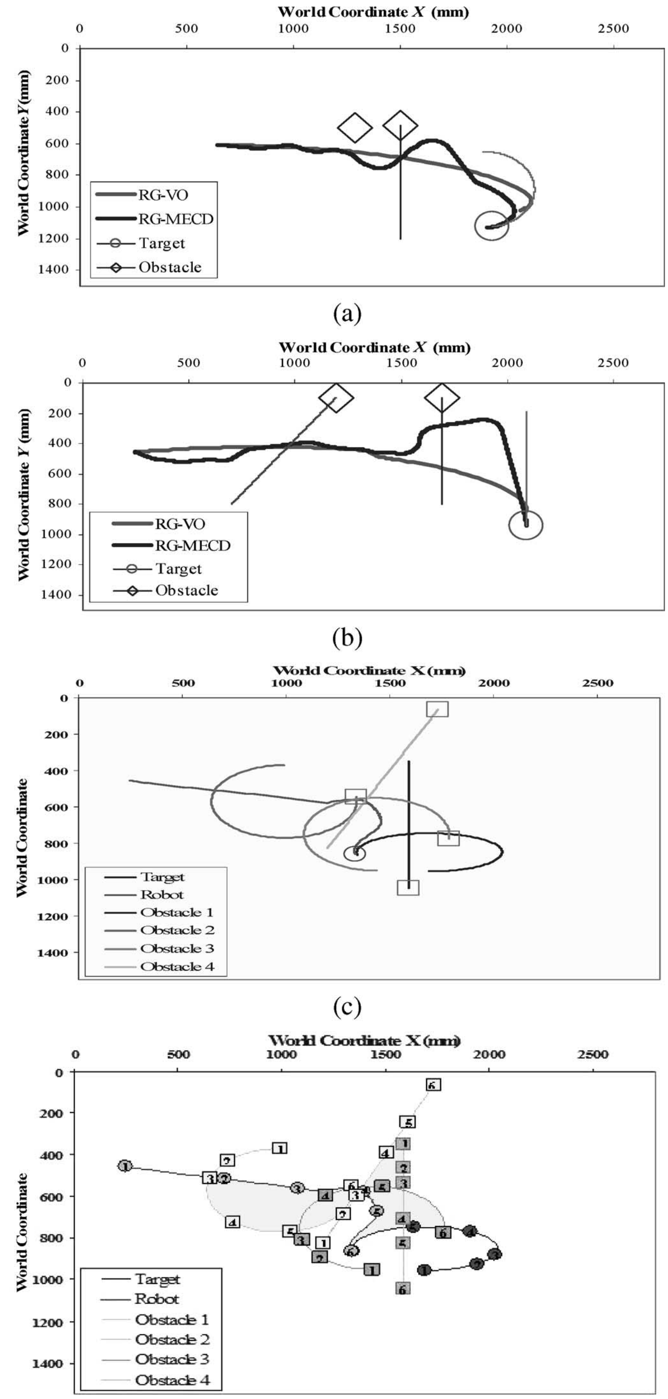


Fig. 9. Simulations. (a) Simulation with one static obstacle. (b) Simulation with two dynamic obstacles. (c) Simulations in a multiple dynamic obstacles complex environment. (d) Time evolution of the avoidance maneuver.

In Fig. 9(d), the dark circles represent the target; the lighter circles represent the interceptor; and the squares represent the obstacles. Each object has a number associated with it ranging from instants 1 to 6, representing the location of the object at that particular point in time. Thus, the initial positions of the objects are represented by the number 1 and the final positions by the number 6. The interceptor

TABLE III
SUMMARY OF THE SIMULATION DATA

S.No	Target Motion		Obstacle Motion		Mean Interception Time(s)		Mean Rendezvous Time(s)	
	Type	Max Vel	Type	Max Vel	RG-MECD	RG-VO	RG-MECD	RG-VO
1	Static	0	Static, Straight, Circular, Sinusoidal	200	6.57	5.80	7.16	6.50
2	Straight	200			6.43	6.03	7.39	6.58
3	Circular	200			5.68	5.05	6.53	5.69
4	Sinusoidal	200			6.19	5.72	8.02	7.09

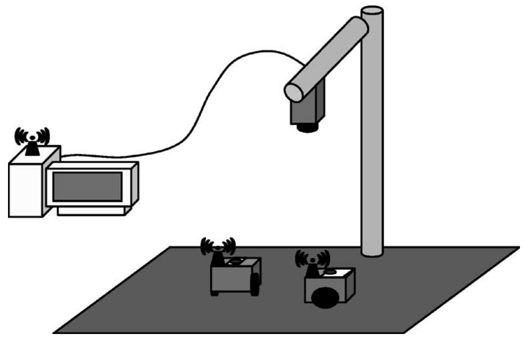


Fig. 10. Physical layout of the setup.

starts by accelerating to avoid the spiraling Obstacle 1 and passes in front of the obstacle (instant 2) and continues rendezvous up to instant 3. Here, the algorithm has to safely negotiate the three obstacles simultaneously: Obstacle 2, which is beneath the interceptor moving in a circular trajectory, Obstacle 3, which is directly in front of the robot moving across the interceptor's path, and Obstacle 4, which is moving straight down. The robot slows down sufficiently to pass in front of Obstacle 2, while allowing Obstacle 3 to pass it. In order to successfully achieve this, the trajectory of the interceptor has to be directed away from the target. From instant 4 to instant 5 the interceptor has to slow down sufficiently to allow Obstacle 4 to pass in front of it, but still avoid Obstacle 2, coming up from behind. After a successful navigation, the interceptor modifies its trajectory for a tail on the rendezvous with the target, which is achieved from instant 5 to instant 6.

In addition to the simulations described above, the proposed method was tested by varying the trajectories and velocities of the target and the obstacles (Table III).

The simulations showed that, when the RG is augmented with the VO, the path of the robot is more time efficient and the motion is mostly linear with lesser demands on the accelerations. The simulations also showed the ability of the algorithm to determine when can time optimality be achieved by accelerating to avoid the obstacle or slowing down in order to maintain the RG time-optimal velocity profile.

V. EXPERIMENTS

The physical layout of the experimental setup is depicted in Fig. 10, and the hardware specifications are given in Table IV. The software for the experiments, running on a Pentium IV 1.6-GHz processor PC, consisted of three modules: image acquisition and processing, trajectory planning, and communication modules. An analog charge-coupled device camera captured the image of the workspace and transferred it to the frame grabber in the PC. The vision algorithm then extracted the positional information of all the objects in the workspace. This information was sent to the trajectory planner, where an acceleration command is calculated for the robot/interceptor. The communication module broadcasted these data to the mobile robots via an RF module connected to the PC through a serial port. The details of the vision system, communication system, and mobile robots are included in the Appendix.

TABLE IV
EXPERIMENTAL HARDWARE

Component	Characteristics
Mobile Robots	RF Controlled
PC	Host computer, frame grabber and RF module
CCD Camera	Resolution: 640 × 480 pixels Lens focal length: 6 mm Distance from floor: 3000 mm
Floor Workspace	2740 × 1500 mm Surface material: Felt

TABLE V
CALCULATION OF THE 95% CONFIDENCE INTERVAL

Mean Improvement	Std Dev	Student t_c (*DOF* N=35)	95% CI	upper limit	lower limit
11.79	**4.66**	**2.03**	**1.58**	**13.36**	**10.21**

TABLE VI
INTERCEPTION DATA EXPERIMENTS

S/NO	Complexity	Obs Vel (mm/s)		Tgt Vel (mm/s)	Mean Interception Time (s)	
		Obs. 1	Obs. 2		ME CD	VO
1	Static Obs- Moving Object	0	0	120	5.29	4.82
2	Moving Obs - Static Object	120	0	0	6.68	6.14

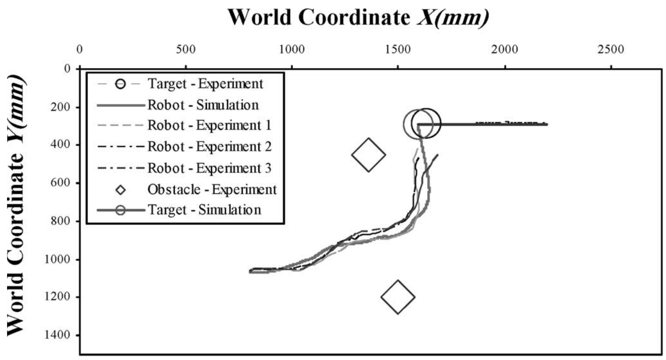
A. *Experimental Results*

The experiments were carried out with the aim to intercept a moving target without trying to rendezvous with it (for equipment safety reasons). Here, again, a comparison is shown with the RG–MECD method. A student t-test analysis of the comparative experimental data was carried out for the 36 experiments performed with the various different initial conditions and trajectories of the target and the obstacles. The results of the analysis are summarized in Table V. This table shows that the mean improvement in the interception time was 11.79. One can conclude with 95% confidence that an improvement of 13.36%–10.21% in the interception time can be obtained by using the RG–VO, when compared to the RG–MECD method.

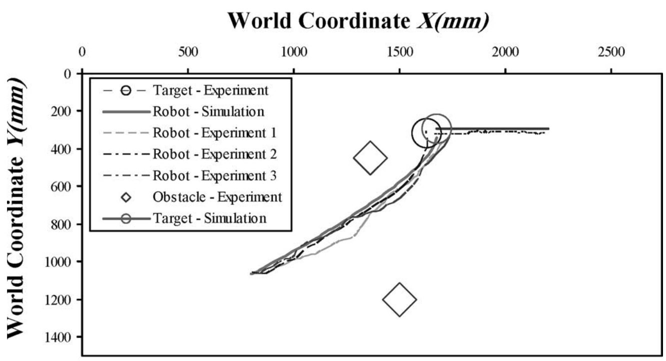
In the first experiment presented here, the object is moving on a straight line, and the obstacles are static; whereas in the second experiment, an obstacle is moving across the path of the robot, and the target is static. Each experiment was repeated three times under identical conditions. The results of the experiments are shown in Table VI and Figs. 11 and 12. The experimental results confirm the ability of our algorithm to maintain a smooth trajectory compared to the one obtained by the RG–MECD for the experiments under the same conditions.

VI. CONCLUSION

A novel RG method is proposed for autonomous robotic interception of moving targets in a dynamic environment with static and/or moving obstacles. The focus has been on two autonomy aspects: 1) time-optimal rendezvous with a moving target and 2) obstacle avoidance. The proposed algorithm uses the RG method to obtain the velocity for the next sampling period, as long as no obstacle avoidance is required. However, in the presence of obstacles, the algorithm uses the VO method, which defines a set of colliding velocities between the

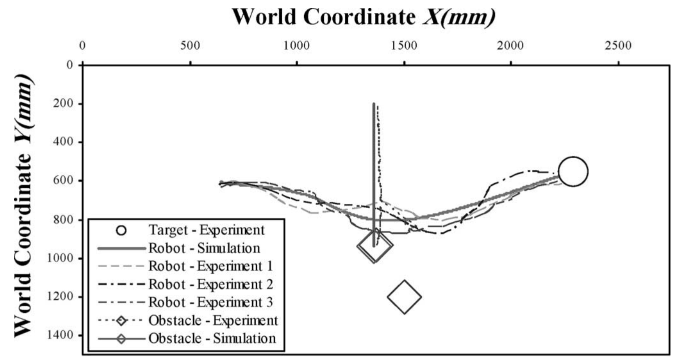


(a)

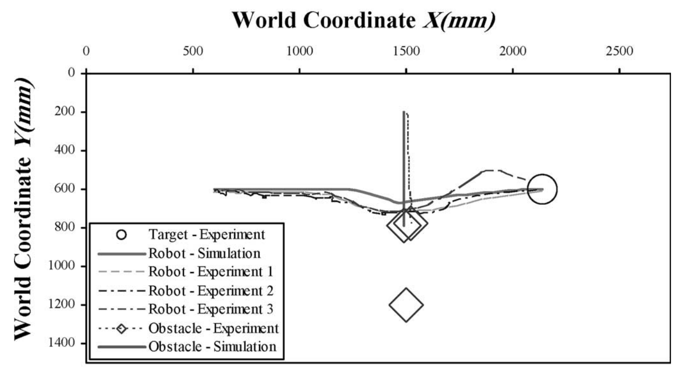


(b)

Fig. 11. Experiments with the static obstacles. (a) RG–MECD and (b) RG–VO.



(a)



(b)

Fig. 12. Experiments with the mobile obstacle. (a) RG–MECD and (b) RG–VO.

robot and the obstacles. By employing a velocity and heading outside the VO, a collision-free trajectory to the target can be ensured.

Furthermore, in our algorithm, instead of using a heuristic search strategy proposed in the original VO method, the search for a feasible velocity for the next sampling interval is reduced to the velocities that are as close as possible to the maximum closing-velocity component obtained from the RG method. For this purpose, a sequence of checks was designed to obtain a feasible velocity for the next sampling period, to keep the deviation necessary for obstacle avoidance minimal from the path obtained by the RG law. The simulations and the experiments have verified the system to be efficient and robust with regard to the interception of the moving targets with various different interception parameters and situations. A drawback of the algorithm is that, in its current implementation, it requires the objects in the environment to remain visible at all times in order to obtain their relative positions.

Appendix
Details of the Experimental Setup

Vision System

The robot, the obstacles, and the target have markers that are color coded for identification. The raw image containing the three channels of data, indicating the intensities of the red (R), green (G), and blue (B) colors in each pixel, is transformed into the YCbCr (luminance, chrominance-blue, and chrominance-red) color space. The transformation is performed by the following equations [31]:

$$Y = 0.299R + 0.587G + 0.114B \tag{A1}$$

$$\mathrm{Cb} = (B - Y)/1.772 \tag{A2}$$

$$\mathrm{Cr} = (R - Y)/1.402 \tag{A3}$$

where Y has a range of [0,255] and Cb and Cr have a range of $[-127.5,127.5]$.

When an image is examined, the weighted Euclidean distances, in the YCbCr color space, between each pixel in the image and the predefined color set are calculated using

$$D = \sqrt{0.15(Y_\mathrm{p} - Y_\mathrm{c})^2 + 0.425(\mathrm{Cb}_\mathrm{p} - \mathrm{Cb}_\mathrm{c})^2 + 0.425(\mathrm{Cr}_\mathrm{p} - \mathrm{Cr}_\mathrm{c})^2} \tag{A4}$$

where D is the weighted Euclidean distance, Y_p, Cb_p, and Cr_p, are the measured YCbCr values of the pixel, and Y_c, Cb_c, and Cr_c are the values of the predefined color set. It was noted that the pixels on the identification marker did not vary more than 18.0 in the weighted Euclidean distance from the defined YCbCr value. This value was, therefore, set as the threshold distance: if a pixel is within this threshold distance of a certain color in the predefined color set, then, it is considered to be that color. After the image has gone through the thresholding operation, the positions of the mobile robot, the obstacles, and the target are determined.

A search is performed to find the markers on all the objects. To achieve the smallest sampling rate, the dimension of the smallest marker is used, denoted here as l pixels. Starting from the pixel location (0,0), every $0.5l$ pixels are sampled along the X and Y directions. If the sampled pixel has the color of the predefined set, a search frame is placed over that pixel. The size of the search frame is twice the diameter of the marker. If the number of pixels of a certain color in the search frame exceeds a predetermined threshold, then, a marker of that color is considered to be located in that search frame. The centroid of that color blob is then calculated to subpixel accuracy using the centroid method [32] (Fig. 13).

With all the markers located, object identification can be performed. The vision program first searches for the blue markers. Once a blue

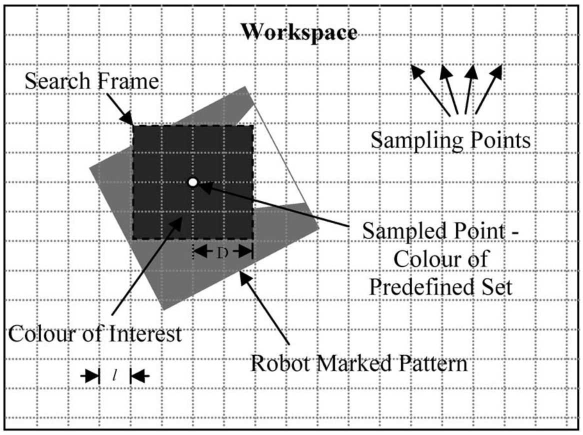


Fig. 13. Color marker search.

marker is found, the algorithm looks for a white marker within a distance of the radius of a robot. If a corresponding white marker is located, then a robot has been successfully identified. The bearing of the object is indicated by an imaginary line drawn between the center of the blue circle to the centroid of the white pattern. The algorithm takes approximately 150 ms to execute (i.e., a frame rate of 6.5 ft/s).

Communication System

The communication system uses wireless transceivers to provide an asynchronous half-duplex link between the host PC and the mobile robots [33]. The transmission baud rate operates at 19.2 kb/s. Two carrier frequencies are available for this system: 866 and 916 MHz. An independent wireless transceiver module is responsible for the communication on the host PC. It is connected to the PC through the RS-232 serial port. The MAX232N chip converts the RS-232 signals into TTL signals, which can then be transmitted by the transceivers. The transceiver modules include the TLP916 and RLP916 for transmitting and receiving, respectively.

Mobile Robots

Two differential-drive mobile robots were used in the implementation of this correspondence: an interceptor and a moving obstacle or target. The mobile robots are powered by two Faulhaber 2842-012C DC motors, rated at 12 V and provide 6.50 W of output power. Maximum torque is rated at 6.88 oz · in. A Faulhaber 38/3 spur gear-head provides a 5.42 : 1 reduction ratio, to provide more torque to the wheels. This gear-head is in turn connected to the wheels through a 1 : 1 ratio gear assembly. Two ball casters provide the balancing support to the robot. The top plate is marked with a colored pattern.

Robot Controller

The robot controllers utilize the Quanser QIC processor core. A baseboard was designed to house this processing unit and other auxiliary modules such as the motor drivers and the wireless transceivers. The controllers are powered by two 7.2-V Canon BP-511 Li-ion batteries. The QIC processor core uses a microchip PIC16F877 microprocessor and has a built-in RS-232 interface. It has 8 K of flash program memory, 10-bit analog-to-digital converter, and a built-in pulsewidth modulation (PWM) controller. The processor is programmed using an embedded C++ language. The baseboard consists of three modules: the transceiver unit, power unit, and motor drivers. The voltage regulators provide a steady 12 and 5 V for the transceiver unit and the motor drivers. The PWM-driven motor drivers used were Allegro Microsystems 3959 series, capable of delivering up to 3.0 A to each motor.

REFERENCES

[1] A. Yilmaz, O. Sami, and R. P. Davis, “Flexible manufacturing systems: Characteristics and assessment,” *Eng. Manage. Int.*, vol. 4, no. 3, pp. 209–212, Sep. 1987.
[2] F. Kunwar, F. Wong, R. Ben Mrad, and B. Benhabib, “Rendezvous guidance for the autonomous interception of moving objects in cluttered environments,” in *Proc. IEEE Conf. Robot. and Autom.*, Barcelona, Spain, Apr. 2005, pp. 3787–3792.
[3] P. Fiorini and Z. Shiller, “Motion planning in dynamic environments using velocity obstacles,” *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 711–727, 1998.
[4] N. A. Shneydor, “Parallel navigation,” in *Missile Guidance and Pursuit*. Chichester, U.K.: Horwood, 1998, pp. 77–99.
[5] H. L. Pastrick, S. M. Seltzer, and M. E. Warren, “Guidance laws for short-range tactical missiles,” *J. Guid., Control Dyn.*, vol. 4, no. 2, pp. 98–108, 1981.
[6] G. M. Anderson, “Comparison of optimal control and differential game intercept missile guidance law,” *AIAA J. Guid. Control*, vol. 4, no. 2, pp. 109–115, Mar. 1981.
[7] D. Ghose, “True proportional navigation with manoeuvring target,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 1, no. 30, pp. 229–237, Jan. 1994.
[8] T. J. Speyer, K. Kim, and M. Tahk, “Passive homing missile guidance law based on new target maneuver models,” *J. Guid.*, vol. 1, no. 13, pp. 803–812, Sep. 1990.
[9] C. D. Yang and C. C. Yang, “A unified approach to proportional navigation,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 33, no. 2, pp. 557–567, Apr. 1997.
[10] P. J. Yuan and S. C. Hsu, “Rendezvous guidance with proportional navigation,” *J. Guid., Control, Dyn.*, vol. 17, no. 2, pp. 409–411, 1993.
[11] M. Guelman, “Guidance for asteroid rendezvous,” *J. Guid., Control, Dyn.*, vol. 14, no. 5, pp. 1080–1083, 1990.
[12] D. L. Jensen, “Kinematics of rendezvous manoeuvres,” *J. Guid.*, vol. 7, no. 3, pp. 307–314, 1984.
[13] H. R. Piccardo and G. Hondered, “A new approach to on-line path planning and generation for robots in non-static environment,” *J. Robot. Auton. Syst.*, vol. 8, no. 3, pp. 187–201, 1991.
[14] M. Mehrandezh, M. N. Sela, R. G. Fenton, and B. Benhabib, “Robotic interception of moving objects using an augmented ideal proportional navigation guidance technique,” *IEEE Trans. Syst., Man, Cybern.*, vol. 30, no. 3, pp. 238–250, May 2000.
[15] J. M. Borg, M. Mehrandezh, R. G. Fenton, and B. Benhabib, “Navigation-guidance-based robotic interception of moving objects in industrial settings,” *J. Intell. Robot. Syst.*, vol. 33, no. 1, pp. 1–23, Jan. 2002.
[16] F. Agah, M. Mehrandezh, R. G. Fenton, and B. Benhabib, “On-line robotic interception planning using rendezvous-guidance technique,” *J. Intell. Robot. Syst.: Theory Appl.*, vol. 40, no. 1, pp. 23–44, May 2004.
[17] T. S. Jacob and S. Micha, “On the ‘Piano Movers’ problem. I: The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers,” *Commun. Pure Appl. Math.*, vol. 36, no. 3, pp. 345–398, 1983.
[18] G. G. Elmer and W. J. Daniel, “Distance functions and their applications to robot path planning in the presence of obstacles,” *IEEE J. Robot. Autom.*, vol. 1, no. 1, pp. 21–30, Mar. 1985.
[19] H. Hu, M. Brady, and P. Probert, “Navigation and control of a mobile robot among moving obstacles,” in *Proc. IEEE Conf. Decision Control*, Brighton, U.K., Dec. 1991, pp. 698–703.
[20] S. Cameron, “Obstacle avoidance and path planning,” *Ind. Robot*, vol. 21, no. 5, pp. 9–14, 1994.
[21] R. Simmons, “The curvature-velocity method for local obstacle avoidance,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Minneapolis, MN, Apr. 1996, pp. 2275–2282.
[22] D. Fox, W. Burgard, and S. Thrun, “The dynamic window approach to collision avoidance,” *IEEE Robot. Autom. Mag.*, vol. 4, no. 1, pp. 23–33, Mar. 1997.
[23] W. Feiten, R. Bauer, and G. Lawitzky, “Robust obstacle avoidance in unknown and cramped environments,” in *Proc. IEEE Int. Conf. Robot. and Autom.*, San Diego, CA, May 1994, pp. 2412–2417.
[24] F. Zhang, A. O’Conner, D. Luebke, and P. S. Krishnaprasad, “The experimental study of curvature-based control laws for obstacle avoidance,” in *Proc. IEEE Int. Conf. Robot. and Autom.*, New Orleans, LA, Apr. 2004, vol. 4, pp. 3849–3854.

[25] N. Y. Ko and R. G. Simmons, "The lane-curvature method for local obstacle avoidance," in *Proc. IEEE Conf. Intell. Robots and Syst.*, Victoria, BC, Canada, Oct. 1998, pp. 1615–1621.
[26] P. Ogren and N. E. Leonard, "A tractable convergent dynamic window approach to obstacle avoidance," in *Proc. IEEE Int. Conf. Intell. Robots and Syst.*, Lausanne, Switzerland, Sep. 2002, vol. 1, pp. 595–600.
[27] H. Hu and M. Brady, "A Bayesian approach to real-time obstacle avoidance for mobile robots," *Auton. Robots*, vol. 1, no. 1, pp. 69–92, Jan. 1994.
[28] D. Fox, W. Burgard, S. Thrun, and A. Cremers, "A hybrid collision avoidance method for mobile robots," in *Proc. IEEE Int. Conf. Robot. and Autom.*, 1998, pp. 1238–1243.
[29] F. Large, S. Sekhavat, Z. Shiller, and C. Laugier, "Towards real-time global motion planning in a dynamic environment using the NLVO concept," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Lausanne, Switzerland, 2002, vol. 1, pp. 607–612.
[30] D. Castro, U. Nunes, and A. Ruano, "Reactive local navigation," in *Proc. IEEE Annu. Conf. Ind. Electron.*, Sevilla, Spain, Nov. 2002, vol. 3, pp. 2427–2432.
[31] D. Bourgin, *Color Space FAQ*. (2004, Aug.). [Online]. Available: http://www.neuro.sfc.keio.ac.jp/~aly /polygon/info/color-space-faq.html
[32] C. B. Bose and J. Amir, "Design of fiducials for accurate registration using machine vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 12, pp. 1196–1200, Dec. 1990.
[33] T. Chan, "Design of control system of a mobile soccer playing robot," M.S. thesis, The Edward S. Rogers Sr. Dept. Elect. Comput. Eng., Univ. Toronto, Toronto, ON, Canada, 2003.