A Delay-Tolerant Potential-Field-Based Network Implementation of an Integrated Navigation System

Rachana Ashok Gupta, Ahmad A. Masoud, *Member, IEEE*, and Mo-Yuen Chow, *Fellow, IEEE*

***Abstract*—Network controllers (NCs) are devices that are capable of converting dynamic, spatially extended, and functionally specialized modules into a taskable goal-oriented group called networked control system. This paper examines the practical aspects of designing and building an NC that uses the Internet as a communication medium. It focuses on finding compatible controller components that can be integrated via a host structure in a manner that makes it possible to network, in real-time, a webcam, an unmanned ground vehicle (UGV), and a remote computer server along with the necessary operator software interface. The aim is to deskill the UGV navigation process and yet maintain a robust performance. The structure of the suggested controller, its components, and the manner in which they are interfaced are described. Thorough experimental results along with performance assessment and comparisons to a previously implemented NC are provided.**

***Index Terms*—Harmonic potential field (HPF), intelligent space (iSpace), network-based integrated navigation system.**

NOMENCLATURE

iSpace	Intelligent Space.
QCFPT	Quadratic curve fitting path tracking.
NCS	Networked control system.
NC	Network controller.
UGV	Unmanned ground vehicle.
HPF	Harmonic potential field.
NBINS	Network based integrated navigation system.
GSM	Gain scheduling middleware.
LoG	Laplacian of Gaussian.
$I(x, y)$	Raw top-view image of the workspace.
ϕ	Harmonic potential field.
$V(x, y)$	Gradient array from HPF.
$R(x, y)$	Reference point corresponding to (x, y).
δL	Discrete look-ahead distance corresponding to one pixel.
G_D	Actual workspace distance in meters corresponding to one pixel in $I(x, y)$.
ν, ω	Linear and angular velocity for the UGV.
$m \times n$	Size of the workspace image I.
$x_a \times y_a$	Actual workspace size in meters.
C_{TM}	Number of computations with template matching.
C_{ED}	Number of computations with edge detection and HPF.
k	Non-real-time computational performance improvement factor.
λ_p	Real-time computational performance improvement factor.

Manuscript received November 12, 2008; revised June 18, 2009. First published July 17, 2009; current version published January 13, 2010.

R. A. Gupta and M.-Y. Chow are with the Advanced Diagnosis, Automation, and Control Laboratory, Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27606 USA (e-mail: ragupta@unity.ncsu.edu; chow@eos.ncsu.edu).

A. A. Masoud is with The Department of Electrical and Computer Engineering, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia (e-mail: masoud@kfupm.edu.sa).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2009.2026764

I. INTRODUCTION

NETWORKED robotics offers a framework for the coordination of complex mobile systems, thus providing the natural common ground for convergence of information processing, communication theory, and control theory. It poses significant challenges, requiring the integration of communication theory, software engineering, distributed sensing, and control into a common framework. Furthermore, robots are unique in that their motion and tasking control systems are colocated with the communication mechanisms. This allows the coexploitation of individual subsystems to provide completely new kinds of networked or distributed autonomy. A physically dissociated navigation system, where the data processing and controller modules could be freely committed and physically located anywhere (NC) has many advantages over a system where the navigation assets have to be physically related to a specific group of UGVs. An NC [1], [21]–[23] allows data to be efficiently shared. It is easy to fuse global information to take intelligent decisions over a large physical space, and it eliminates unnecessary wiring. It is also scalable because it is easy to add more sensors and UGVs with very little cost and without heavy structural changes to the whole system. Most importantly, an NC connects virtual space to physical space, making task execution from a distance easily accessible (a form of telepresence). With the advances in computer and communication technologies, these systems are progressively becoming easier to build.

One of the modes an NC could operate in concerns the integration of the assets that a robotics agent needs for operation. In this case, the bulk of the intelligence and processing needed for the robot to function is placed on a remote server that receives data of the environment of the robot and transmits guidance signals to the robot. If properly designed, this layer may be used as an element for building another layer of networking. The higher layer is concerned with networking multirobots so they will be able to share a common environment and even cooperate to collectively perform a certain task, e.g., RoboCup [46].

In order to actualize an NC that has certain capabilities, both the components and the networking scheme have to give rise to

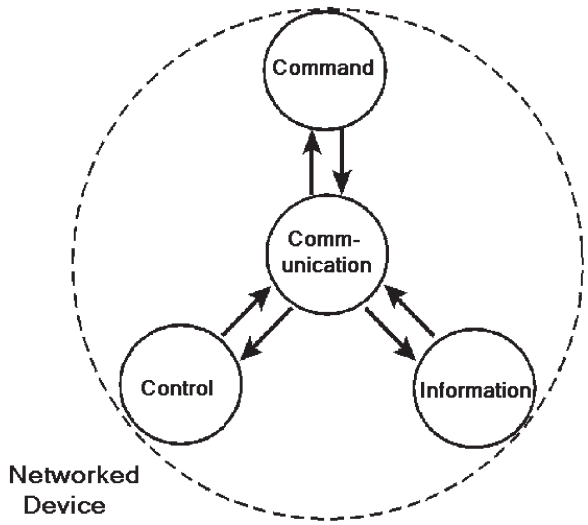


Fig. 1. Basic functions in networked devices.

four basis functions which form the substrate of the capabilities that an NCS is required to project (Fig. 1). These basis functions are information acquisition, command, communication, and control (C3I).

A navigation controller that configures these modules performs tasks independently yet together, hence integrating them into one system. There is a wealth of techniques available for actualizing each one of the basis function modules. However, the norm seems to be developing each module in isolation of the others by either placing, most probably restrictive, assumptions on the modules that can function with the one being developed or assuming altogether, without providing a proof, that suitable modules do exist. An integrated view to the functioning of NCS that tackles design at the system level is still much in need.

A wide branch in literature [9] focuses on the different control strategies and kinematics of the actuators/vehicles [20], [32]. Another research area concerning NCSs studies the required network structure [47], [48] according to the network delays observed, packet losses, and suitable protocols for control [33]. Techniques specifically focusing on navigation and path planning for autonomous robots using a predefined environment and obstacle models have also been an important subject of research for the past few decades. For example, the fast-marching (FM) method by Sethian [26], the potential field approach by Khatib [3]–[15], and the fuzzy-logic-based navigation system by Yen and Pfluger [34] are some of the popular techniques used for unmanned vehicle navigation [30], [31]. Geometrical techniques like occupancy grids [35], topological mapping [36], and Voronoi diagrams using sonar, laser, and visual sensors [15] are some of the techniques used to model the environment before path planning. While studying all these modules separately, it is highly unlikely to find a realistic command module that jointly takes into consideration the realization of an admissible control signal when converting a task and constraints on behavior into a group of reference guidance signals (a plan). There are few system architectures or middleware developed to put together a heterogeneous system [29], [37]. These architectures focus mainly on the modularization and abstraction of the system to achieve flexibility and scalability in design rather than on developing procedures and guidelines for selecting the controller components to enhance the efficiency of each module separately as well as improving system performance as a whole. The same thing may be said about the information acquisition module. Many systems and algorithms such as template matching [25] and color recognition [39] have been developed using visual and other local sensing techniques to control ground and aerial vehicles autonomously. However, the suitability of the environment representation for use with the communication and command modules is rarely taken into consideration although it is the key point in any practical application of an NCS. The individualistic approach to design is highly likely to make an NCS overly expensive and complex. Furthermore, incompatibilities among the components could give rise to disruptive hidden modes of behavior and seriously limit system performance. In [38], Brooks studied the requirements for an agent or a device to have a reasonable chance of success operating in a realistic environment. He found that a device of such a sort must be intelligent to enable it to deal with novel situations it encounters, which the designer did not initially take into consideration. Moreover, it must be situated as an integral part of the environment, it must be embodied, and its behavior must be emergent.

In this paper, modules for realizing the basis functions needed for generating intelligent behavior are carefully selected and interlinked to realize a demonstrably efficient visually servoed networked controller for a differential drive UGV. The controller uses the Internet as a communication medium tying the data stream from an overhead camera to the processing and decision making unit that lies on a remote server back to the end-effector UGV. Although the controller focuses on networking resources needed by one UGV to operate, its design and the choice of modules yield a social controller that allows other UGVs using the same type of controller to share the same workspace (see Section VI-F). The controller shows good robustness in the presence of time delay. This, among other things, makes it possible to accommodate the channel coding process needed to guard against the following:

1) insertion of data in the communication channel;
2) eavesdropping;
3) denial of message delivery by the recipient;
4) data-quality degradation caused by noise and other artifacts.

Total cutoff of the guidance signal that is caused by the unavailability of the network can only be dealt with by the UGV reverting to the safe state of obstacle avoidance described in [2].

A major feature of the controller is the removal of the computationally expensive and error-prone image vision and interpretation module. The controller relies only on the raw output of a low-level-vision edge-detection module to directly feed the planner. This provides the NC with the ability to handle image streams containing complex unstructured components in a timely manner. Moreover, the planning and control modules are interfaced so that the NC exhibits high resistance to the unstructured and random delay Internet induced in the visual servo loop. The modules are selected and networked in a manner that observes Brooks guidelines and minimizes intermodule

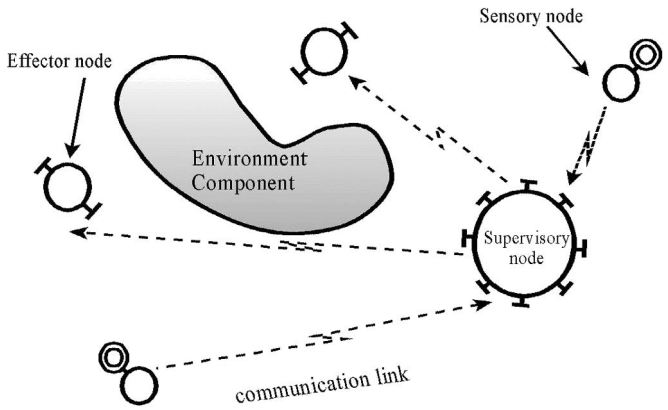


Fig. 2. NCS basic layout.

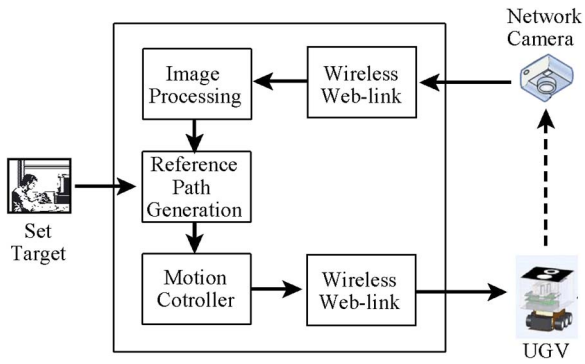


Fig. 3. Prototype of the NBINS used as a test bed—iSpace.

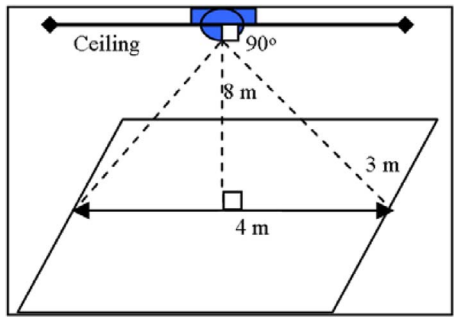


Fig. 4. Single-camera setup for the global workspace image.

incompatibility and conflict. Although there are no proofs that such guidelines can tackle the design of a general NC, the authors found them useful in designing the NC suggested in this paper. Experimental observations show that the networked modules synergetically interact to yield an NC with an intelligent, robust, emergent, and situated behavior.

The organization of this paper is as follows. Section II explains the NCS setup, i.e., the design guidelines followed by the structural description of the test bed iSpace which is used to implement the NC. Section III discusses the vision and planning modules used in building the NC along with the manner in which they are interfaced. In Section IV, the interface between the planning and the control modules is discussed. Section V compares the performance of the suggested NC to a previously implemented NC on the same test bed. Thorough experimental results of the suggested NC are given in Section VI, and conclusions are placed in Section VII.

II. NC Setup and Design Guidelines

This paper is the start of an effort to build a distributed resource NCS that provides a casual operator with means to effectively manage a group of mobile effector nodes that are operating in a static cluttered environment (Fig. 2).

In its fully developed form, it should be possible to deploy the effector nodes of the NCS to perform a variety of tasks that are set by a supervisory node. The components of the controller should be able to use different types of media to communicate among themselves or with the supervisory node. The components should also have adjustable autonomy with a behavior ranging from strict teleoperation by the supervisor based on an *a priori* available plan to a pure on-the-fly goal-driven self-organization. The system should respond to unfavorable circumstances such as loss of assets with graceful degradation in capabilities and performance instead of a total failure to carry out the intended task.

An NCS prototype which we hope to be a starting point for building such a system is suggested in this paper. The proposed NCS is an intelligent Web-based visual servo system with a user/operator interface (Fig. 3). It consists of a physical and a virtual environment. The physical environment is divided into two parts. The first part is the workspace environment which consists of a differential drive UGV operating in a stationary irregular clutter, a network camera recording the UGV, and the workspace. The second part is the NC environment which consists of a communication channel and equipment, interface, and data processing devices. The virtual environment consists of specialized software modules whose function is to construct a virtual representation of the environment (one is operator centered, and the other is planner centered), command and decision making, reference generation, and control.

The NCS test bed used here is called iSpace. iSpace is used to implement different algorithms and control strategies to compare and analyze the NCS performance. The iSpace platform [24] has the following components.

1) An overhead network camera used as a global sensor to pass the visual information about the workspace to the main controller. Currently, this is the only source of sensory data that the system is utilizing. Future work will consider using feed from local sensors onboard the UGV along with that from the webcam to enhance performance. The webcam collects the top-view image of the 2-D space of interest and sends it to the main controller for processing. The camera is mounted on the ceiling of the room (the indoor workspace) facing downward, capturing the 90° top view of the workspace, as shown in the Fig. 4 (details are in Table I).
2) The NC with graphical user interface (GUI). The NC can be on any remote computer in the world with Internet access. The main NC is connected to the global sensor and the actuator through a wireless channel. All modules and intelligence are thus located on the NC to process the top-view image of the workspace from the network

TABLE I
CAMERA IMPLEMENTATION DETAILS AND SPECIFICATIONS

Camera tilt angle	0
Workspace size per camera view (x_a x y_a)	3m x 4m
Size of the workspace image (m x n)	320 x 240
Camera height	8 m
Camera image transfer rate (average)	5 images /s
Image size (average)	6KB

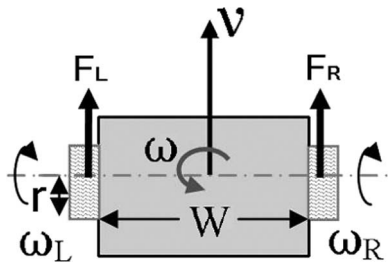


Fig. 5. Model of a differential drive UGV.

camera and generate the mobility control signal to be communicated to the UGV

$$\begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} = \begin{bmatrix} 1/r & +W/2r \\ 1/r & -W/2r \end{bmatrix} \begin{bmatrix} \nu \\ \omega \end{bmatrix}. \tag{1}$$

3) A differential drive UGV (Fig. 5). The UGV does not have any local sensors onboard. It only receives linear speed (ν) and angular speed (ω) from the NC on the wireless channel to control the motion. Equation (1) is implemented in the UGV controller to drive the wheel motors.
4) The network used for communication is the Internet. All the components are connected to the NC over the 802.11b wireless channel. All the sensors and actuators at one location can be on LAN which is in turn connected to the Internet for remote operations. The information and the control commands use User Datagram Protocol (UDP)/IP for communication. UDP is less complex than Transmission Control Protocol and saves the time required for the packet acknowledgment. This is an important required aspect for time-sensitive systems since the control information carried by the dropped packet becomes obsolete to send again. As the delay on the actual network cannot be controlled or reproduced, a network delay generator (buffer implementation) is implemented for experimental purposes such that it can be inserted in the loop instead of the actual network to generate the desired network delay in the system. Although it is not always the case that the dropped packets become obsolete to send again (retransmissions, mainly with slow sampling systems, can be worth doing since they allow recovering the state of the system much faster than waiting for the next sample), there is no purpose in trying to retransmit a packet beyond a given packet deadline.

In this paper, we use the physical edges of the image stream from the camera as the source of information to represent the contents of the workspace. The HPF approach [3], [4] to motion planning is used for generating the guidance signal for the UGV. The QCFPT controller suggested in [20] is used for converting the guidance signal into a control signal that the UGV may use for reaching a target point in the workspace.

iSpace is used as a test bed to present the experimental results of this paper. It is also used to implement different algorithms and control strategies to compare and analyze their performance. We use edge detection to extract the contents of the workspace, HPF for guidance, and a quadratic curve "path-tracking" controller in [20] to generate the control signal. The GSM delay compensation module [22], [23] is also used with the QCFPT. It is shown in the following that the aforementioned components can be networked together to yield a demonstrably efficient NC. This particular choice of modules to realize the basis functions needed by an NC yields a compatible set of modules that satisfy the following conditions.

1) The format of the representation used for the workspace captures necessary and sufficient information about the environment for UGV navigation. It also has minimal size in order not to congest the communication channel.
2) Each module exhibits flexibility in terms of avoiding restrictive assumptions on the type of data it is required to process.
3) The format of the output from each module is compatible with that of the next module, making it possible to practically accept the input in a raw form with almost a negligible amount of processing.
4) The modules collectively exhibit good resistance to artifacts that the NCS may encounter such as delays in communication and processing as well as sensor noise.

We believe that these criteria are the key to building a system that yields satisfactory performance. The features possessed by the modules and the advantages in putting them together are discussed and demonstrated later in this paper.

III. VISION AND PLANNING MODULES

In this section, a vision module utilizing a simple edge detector and the planning module which uses the HPF approach along with their interdependence are discussed.

A. Describing Contents of iSpace Using Edge Detection

Edge detection is a classic early vision tool used to extract discontinuities from the image as features. All the discontinuities which are more or less obstacle boundaries are represented by binary edges in the edge-detected image of the UGV workspace. The edge-detected image of the workspace is referred to as the edge map. An edge map may be described as

$$\begin{aligned} E(x_i, y_i) &= 1, \qquad \text{if } (x_i, y_i) \in \Gamma \\ &= 0, \qquad \text{if } (x_i, y_i) \notin \Gamma \; \forall \, (i, j) \end{aligned} \tag{2}$$

where $E(x_i, y_i)$ is the image representing the edge map and is the set of boundary points for all obstacles in the workspace. There are several edge-detection techniques available in the literature. We decided to use the modified LoG detector [18], [19]. It was proven in [18] that an LoG detector with a noise

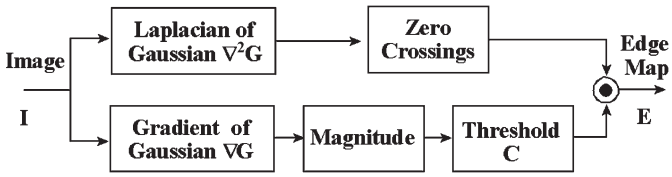


Fig. 6. Complete edge detector block diagram.

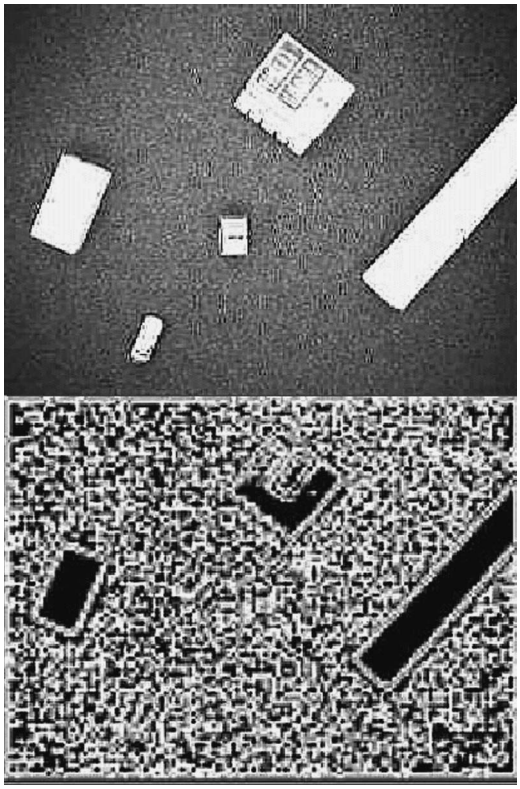


Fig. 7. Workspace image I and its zero-crossing contour.

removal mechanism has many attractive properties. Although simple, it can be made robust when dealing with natural unstructured scenes having widely varying contrasts. The edge detector used here has two branches (Fig. 6). One branch processes the workspace image I with a circularly symmetric approximated LoG operator to estimate the second derivative of the image. The zero-crossing contours of this estimate are used to construct the potential edge map of the image

$$G(\sigma, x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right) \tag{3}$$

$$\nabla^2 G(x,y) = \frac{\partial G}{\partial x^2} + \frac{\partial G}{\partial y^2}$$

$$I_{\mathrm{LoG}} = I \otimes \nabla^2 G(x,y). \tag{4}$$

Such a map is extremely noisy and practically useless as shown in Fig. 7 unless the contours corresponding to the physical underlying edges are appropriately filtered out [18].

The filtering adopted here is based on the strength of the edge, i.e., its contrast. If the contrast of an edge is above a certain threshold ζ, the edge is accepted as authentic; otherwise, it is rejected as a noise-induced artifact. Therefore, the second

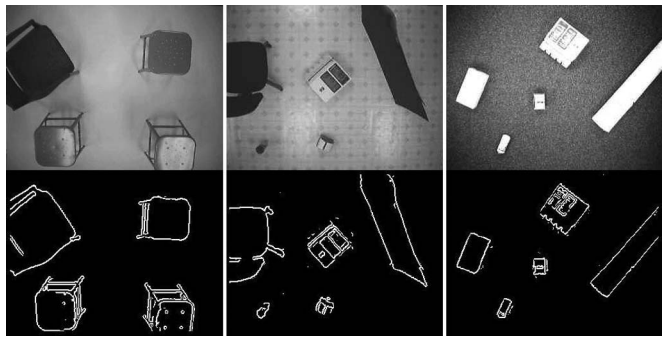


Fig. 8. Output of the edge detector in Fig. 6.

branch for the edge-detection block is an approximated first-derivative gradient operator (5) of 2-D Gaussian function (3). It is used to produce an estimate of the contrasts

$$\nabla G(x,y) = \left[\frac{\partial G}{\partial x^2} \;\; \frac{\partial G}{\partial y^2}\right]^{\mathrm{T}}$$

$$I_{\mathrm{GoG}} = I \otimes \nabla G(x,y). \tag{5}$$

Fig. 8 shows a few sample images from the actual iSpace setup with different backgrounds and obstacles along with their corresponding edge maps $E(x,y)$.

B. Two-Dimensional HPF Planner for Path Planning

The harmonic function on a domain $\Omega \subset R^2$ for a continuously differentiable ϕ is a function which satisfies the following Laplace equation:

$$\nabla^2 \phi = \frac{\partial \phi}{\partial x^2} + \frac{\partial \phi}{\partial y^2} = 0. \tag{6}$$

The use of potential field in motion planning was introduced by Khatib [2]. The use of harmonic functions for planning was suggested almost two decades ago [3], [4] to generate a navigation path in robot workspace, avoiding the spurious local minima in Khatib's settings. HPFs have proven themselves to be effective tools for inducing in an agent an intelligent, emergent, embodied, context-sensitive, and goal-oriented behavior (i.e., a planning action). A planning action generated by an HPF-based planner can operate in an informationally open (i.e., the planner does accept external information during the task execution phase) and organizationally closed (i.e., the planner does not accept external assistance in making decisions) mode [10], enabling an agent to make decisions on the fly using online sensory data without relying on the help of an external agent. HPF-based planners can also operate in an informationally closed (i.e., the planner does not accept external information during the task execution phase) organizationally open (i.e., the planner does accept external assistance in making decisions) mode which makes it possible to utilize existing data about the environment in generating the planning action as well as elicit the help of external agents. A hybrid of the two modes may also be constructed. In [11], vector HPFs were used for planning with robots having second-order dynamics. In [7], the HPF approach was modified to incorporate joint constraints on

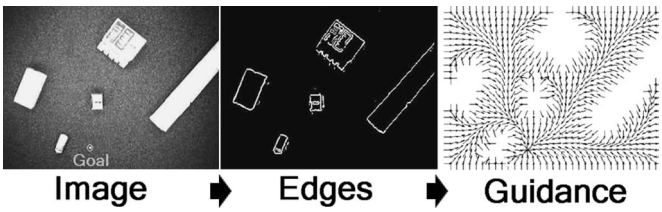


Fig. 9. HPF-edge detector interface.

regional avoidance and direction. The decentralized multiagent planning case was tackled using the HPF approach in [41]. A navigation HPF can be generated using both analog and digital VLSI chips [43]–[45].

A basic setting for generating the HPF from an edge map is

$$\begin{aligned} & \nabla^2\phi(x,y)=0, && x,y\in\Omega \\ \text{subject to} \quad & \phi(x,y)=1, && (x,y)\in\Gamma \\ & \phi(x,y)=0, && (x,y)=(x_T,y_T) \end{aligned} \tag{7}$$

where ∇^2 is the Laplace operator, Ω is the workspace of the UGV, $(\Omega \subset R^2)$ is the boundary of the obstacles, and (x_T, y_T) is the target point. The obstacle free path to the target is generated by traversing the negative gradient of ϕ

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = -\nabla\varphi(x,y). \tag{8}$$

Convergence to the target point and avoidance of hazardous regions were proven in [7].

Fig. 9 shows the result of the HPF planner after applying it to the edge maps of actual workspace images in iSpace. The arrows represent the negative gradient direction at each point converging to the destination point. Thus, HPF converts the edge map into a region-to-point guidance function by accepting the edge data without any processing.

As mentioned in [3], a potential function in any connected component (within the closed boundaries) without a goal point will converge to a constant potential in that UGV configuration space, hence leading to a vanishing guidance field. It is worth noting that all the connected components in the edge map represent the closed obstacle boundary. The successive relaxation method used in [3] to find the HPF is computationally efficient. The fixed number of iterations for a given workspace size and grid resolution makes the algorithm complexity $O(k)$. As discussed in [40], HPF itself is a probabilistic measure of hitting the obstacles. As can be seen from the results obtained, the successful integration of the LoG edge detector and the HPF planning module is achieved in a manner that uses minimal sufficient information needed to safely guide the robot to its target.

IV. Planner–Controller Modules

In this section, the interconnection between the HPF planning module and the path-tracking module is described.

A QCFPT controller is implemented as the motion controller for the UGV to traverse the path generated by the planner from the source to the destination point. The basic principle of this control algorithm explained in [20] by Yoshizawa *et al.* is briefly described. A reference point is moved along a desired path so that the length between the reference point and the UGV is kept in some distance (d_0). Control (velocity) commands—linear speed (ν, in centimeters per second) and angular speed (ω, in radians per second)—are generated for the UGV to reach that reference position from the current position. The original algorithm was suggested as a solution to the path-tracking problem. The path-tracking controller is always trying to keep the UGV on the reference path by calculating the reference points along the path irrespective of the UGV's current position and distance from the reference path. The only input that the path-tracking controller requires in each control loop iteration is the current coordinates of the UGV (x_c, y_c, θ_c) and the reference point coordinates $(x_{\text{ref}}, y_{\text{ref}})$. In its current form, this controller operates in a path-tracking mode. In the following, we describe an interconnection method with the HPF module that allows this controller to operate in the robust and efficient "goal-seeking" mode.

A. HPF Planner and Motion Controller Integration

The QCFPT controller is used so far in cases where the path to be tracked is already known. However, the gradient array of HPF has features that make it possible to use this method more efficiently without *a priori* knowing or calculating the path to be tracked. The 2-D gradient matrix from an HPF contains all the directional information required for the UGV to reach the goal from any point in the workspace (Fig. 9). In other words, every point in the region is associated with a guidance vector that is directing it toward the next reference position leading toward the goal. This can be treated as the same reference position required by the quadratic curve controller. Thus, the important region-to-point guidance property of HPF makes it an efficient path planner to be combined with the quadratic controller without a reference path generation.

We need to compute the gradient array (V). V_x and V_y are the x- and y-components of $-\nabla\phi$ (negative gradient), respectively, as described in (9). The reference position (x_R, y_R) for each current position (x, y) is calculated from the gradient array of the HPF $(\nabla\phi)$. Then, each reference position $R(x_0, y_0) = (x_R, y_R)$ is calculated as

$$V_x = \frac{-\frac{\partial\phi}{\partial x}}{\sqrt{\left(\frac{\partial\phi}{\partial x}\right)^2+\left(\frac{\partial\phi}{\partial y}\right)^2}} \quad V_y = \frac{-\frac{\partial\phi}{\partial y}}{\sqrt{\left(\frac{\partial\phi}{\partial x}\right)^2+\left(\frac{\partial\phi}{\partial y}\right)^2}} \tag{9}$$

$$x_R = x_0 + V_x(x_0,y_0) \quad y_R = y_0 + V_y(x_0,y_0). \tag{10}$$

We choose the reference position at a look-ahead distance (d_0) from the current position as shown in Fig. 10(a). Let the discrete look-ahead distance be represented by δL in pixels. If $\delta L = 2$, then, for the current position (x_0, y_0), $R(x_0, y_0) = (x_2, y_2)$, as shown in Fig. 10(b). Thus, for $(\delta L > 1)$, (10) is calculated in a loop of δL iterations to get a reference point at a distance δL.

The accuracy of the path traced will be sensitive to d_0 as the quadratic curve controller will do curve fitting $(y = Ax^2)$ between the current (x_0, y_0) and the reference point (x_R, y_R). The magnitude of the linear speed and the angular speed is proportional to d_0 between (x_0, y_0) and (x_R, y_R). A small

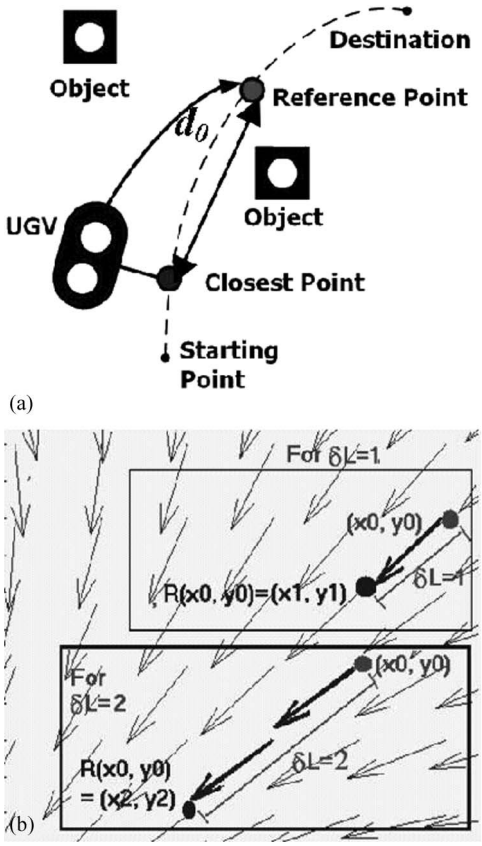


Fig. 10. (a) Path-tracking control. (b) HPF gradient directional array.

δL will keep the UGV trajectory close to the desired path. However, the time to complete the path will increase because of the small ν and ω. With large δL (e.g., $\delta L = 6$), the quadratic curve controller will approximate the path between (x_0, y_0) and (x_6, y_6), which may not match the exact path generated by the HPF planner going through $\{(x_0, y_0), (x_1, y_1), \ldots, (x_6, y_6)\}$. Thus, distance error increases, and the probability of hitting nearby obstacles increases particularly in the case of large network delays (Figs. 11–13).

As described in [20], d_0 is calculated depending upon the curvature of the path (A) at that current position by (11a). Thus, δL should be proportional to d_0 (11a). Then, ν and ω are calculated using (11b). To choose the reference point to start with, we take δL as its minimum value, i.e., $\delta L = 1$. Thus, the error vector (e_x, e_y, e_θ) is calculated for (x_c, y_c, θ_c) and $(x_{\text{ref}}, y_{\text{ref}})$. To fit a curve

$$y = Ax^2 \qquad A = \text{sign}(e_x)\frac{e_y}{e_x^2}$$

$$d_0 = \frac{d_{\max}}{1 + \beta\lfloor A \rfloor} \qquad \delta L = \left\lfloor \frac{d_0}{G_D} \right\rfloor \tag{11a}$$

$$K_n = \text{sign}(e_x)\frac{\alpha}{1 + |A|}$$

$$\nu = K_n \qquad \omega = 2A \cdot K_n. \tag{11b}$$

G_D is a constant representing workspace resolution in meters per pixel. $G_D = 4/320 = 3/240 = 0.0125$ m with workspace image resolution of 320 × 240, the area covered by the network camera, i.e., workspace size is 4 m × 3 m and β is a positive constant that the operator may use for tuning (here, $\beta = 1$). The aforementioned equations yield a δL that is inversely proportional to the curvature of the path. A high curvature yields a small δL and a low UGV speed, while a straight path with a small curvature yields a high δL and a high UGV speed. Thus, dynamically choosing the look-ahead distance will deal a tradeoff between the path-tracking accuracy and the time required to reach the goal. Fig. 14 shows the test results with dynamic look-ahead distance. A performance improvement in both the time to reach the destination and the tracking error with respect to the ideal path is accomplished compared with the fixed look-ahead distance results in Fig. 11. The flowchart describing the integration of the modules on the iSpace platform is shown in Fig. 15.

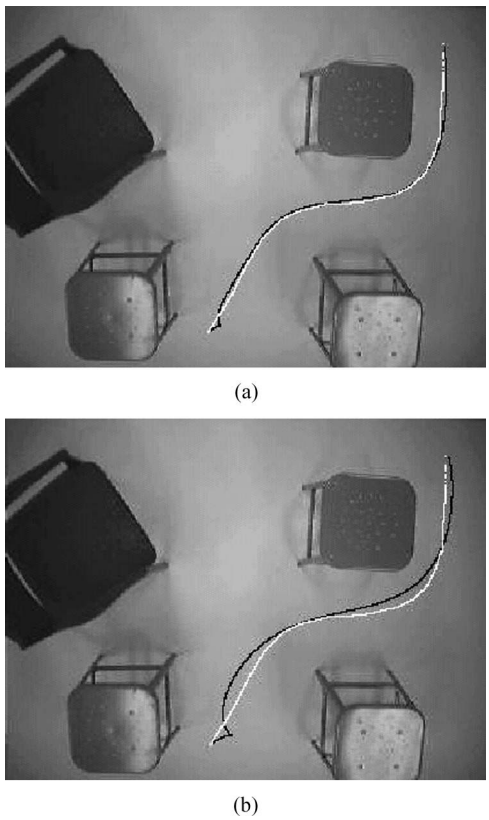


Fig. 11. Effect of different values of look-ahead distances chosen. T is the total time to reach the destination. (a) $\delta L = 1$, $Network\ delay = 0.1$ ms, and $T = 27$ s. (b) $\delta L = 8$, $Network\ delay = 0.1$ ms, and $T = 16$ s.

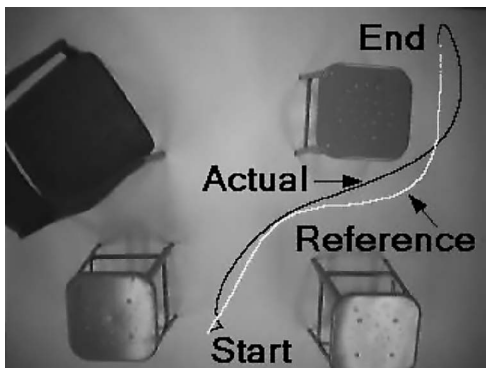


Fig. 12. Large look-ahead distances with high network delay.

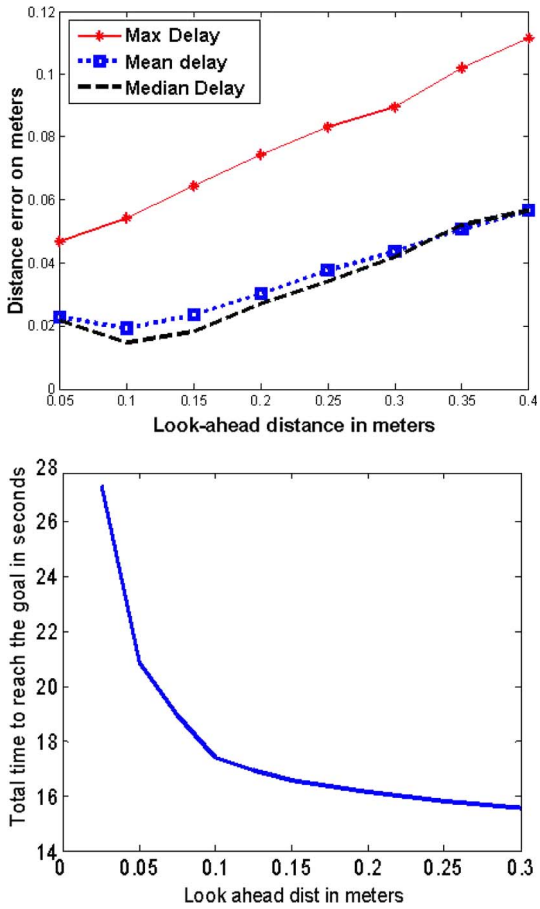


Fig. 13. Distance error increases and total time required to reach the goal decreases with an increase in δL.

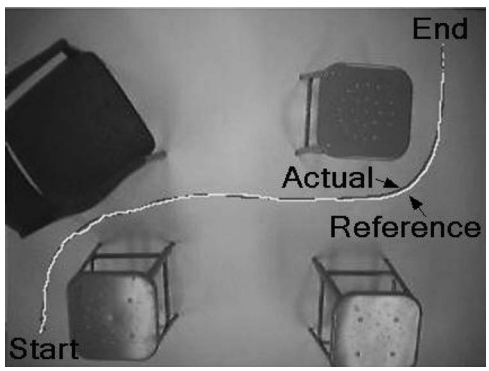


Fig. 14. Test result with dynamic look-ahead distance.

V. Comparison

The important modules in the iSpace are obstacle registration, path planning, motion controller modules for the UGV, and the network delay compensator [22], [23]. While we implemented and presented results with the edge detection and the HPF-based structure for the integrated system, we also implemented the same NC with the popular template-matching-based obstacle recognition and FM-based path-planning modules integrated with the same QCFPT controller. The following part of this section describes the implementation of template-matching and FM methods in iSpace and its comparison with

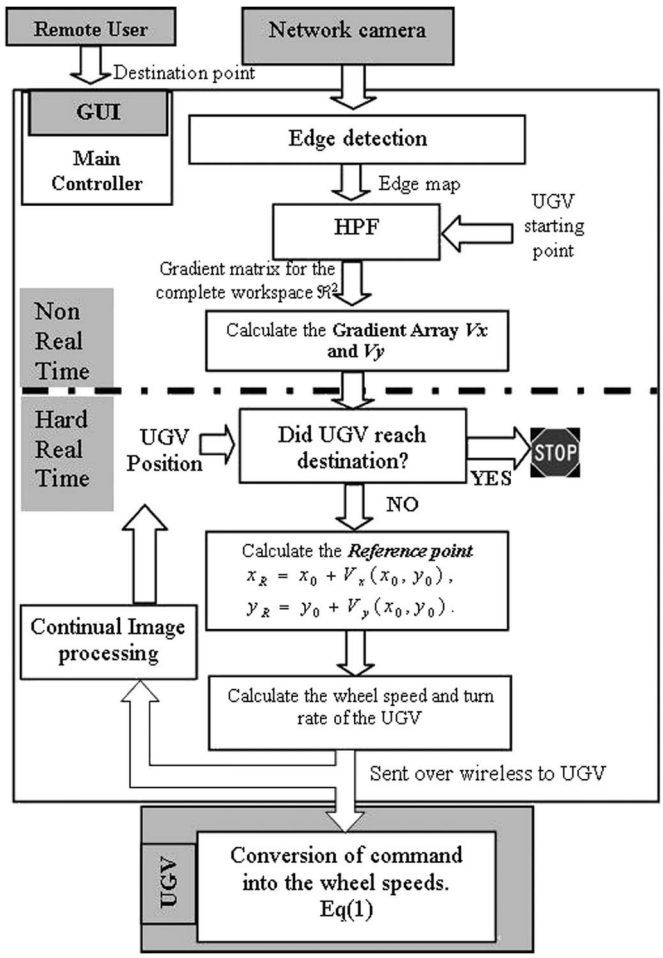


Fig. 15. Complete flow diagram of the network-based integrated system with edge detection and HPF planner.

the edge detection and HPF-based structure. We compare the previous modules with the proposed ones and examine the performance improvement in iSpace as a whole.

A. Template Matching for Content Recognition

Template matching can be used to extract the information from the network camera image to know the location of the obstacles and the position and orientation of the UGV in the workspace. In this case, we pasted circular templates on top of obstacles for recognition (circles are invariant to rotations). The camera is kept at a known constant height h over the workspace. The size of the templates on the top of the UGV and the obstacles are also kept constant, allowing the use of a template image with a fixed resolution (20×20) size, i.e., no zoom factor is considered as it increased the computations required by n-fold given that n is the number of zoomed images considered for matching. The NC does not have any information about the size of the obstacle. Therefore, we draw a circular safety margin of radius r_{safe} around the obstacle. The area inside the safety margin is considered as part of the obstacle by the path-planning module. Details about the template-matching algorithm implemented in iSpace may be found in [24].

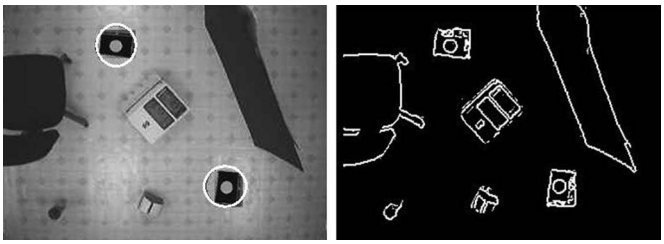


Fig. 16. Template-matching and edge-detection outputs for the same workspace image.

B. Comparison of Edge Detection With Template Matching

Although the use of template matching is a standard practice in visual servo systems, there are many inherent factors in this approach that could limit performance. Some of these factors are the following.

1) Restrictions on the contents of the environment. The obstacles recognized by template matching have to be a part of an *a priori* known set of obstacles with predefined size, shape, color, and templates. These restrictions are removed by using edge detection.
2) Definite knowledge of boundaries of obstacles. With template matching, the main controller only knows the location of the center of the template on the obstacle without having any quantitative and definite knowledge about the size and shape of the obstacle required for successful and guaranteed obstacle avoidance. The edge detector marks the boundaries distinctively. Thus, the path planner directly receives from the edge detector better information about the contents of the 2-D space of the UGV (Fig. 16).
3) Computational complexity. The major parts of the calculations involved in both template-matching and edge-detection algorithms are the convolution with the template image and the convolution of the edge kernel with the black and white workspace image. The size of the template image is 20×20, and the edge kernel size is 7×7

$$\begin{aligned} C_{\mathrm{TM}} &= ((320-20) \times (240-20)) \times (20 \times 20) \\ &= 26\,400\,000 \\ C_{\mathrm{ED}} &= (320 - 7 \times 240 - 7) \times (7 \times 7) \\ &= 3\,573\,521. \end{aligned}$$

The factor k by which edge detection is faster than the template matching can be estimated as shown in

$$k = \frac{C_{\mathrm{TM}}}{C_{\mathrm{ED}}} = \frac{264\,000\,000}{3\,573\,521} \approx 7.3. \tag{12}$$

C. Path Planning Using FM Method

The input to the path generation modules is the position and orientation of the UGV (x_u, y_u, θ_u) and the obstacles' positions received from the template-matching module and the UGV's destination (x_d, y_d) from the GUI set by the user. The output of this module is an obstacle-free path composed of a series of points from the initial UGV position to the destination. A Dijkstra-like method named the FM method was applied to solve the optimal path generation problem [26].

D. FM Path Planner Versus HPF Path Planner

1) Problem description. The reference path calculated by the FM method is a function of the source point, destination point, and obstacle map. The point of consideration is that, being an NCS (e.g., iSpace), the UGV might wander off from the path in due course of movement due to network delay. In such case, it might prove practical and efficient for the UGV to go to the next reference position which might not lie on the previously calculated reference path. Thus, once the UGV is not on the path, there is no guarantee that the quadratic curve fitted between the UGV (off the path) and the reference position (on the path) considers the obstacle avoidance. This creates a requirement for each point in space to have a reference point toward the destination which will have built-in knowledge of the obstacle boundary presence instead of a reference path. With the HPF planner, the workspace surface image is converted into a region-to-point guidance function that is dependent on the destination point and the (obstacle map) edge map. Thus, each point in the region has a reference point associated with it toward the destination and avoids the obstacle by making sure that quadratic curve fitted between the UGV (anywhere in the region) and its reference point is avoiding the nearby obstacle.

   Although the reference point is chosen on the path [Fig. 17(a)], it increases the probability of hitting the corner of the obstacle. The reason is that the proximity of the obstacle was not being considered. In Fig. 17(b), the reference point is chosen at the same distance from the current position, but with the current position being near the obstacle, the Dirichlet's setting generates the gradient value driving the UGV away from the obstacle. Since the reference point is calculated from the gradient point, it is naturally pushed away from the obstacle.
2) The complexity of the FM is $O(N \cdot \log N)$. On the other hand, HPF consists of a simple averaging operation on the edge array for a fixed number of times (for example, M), therefore making it an $O(M)$ algorithm. This comparison clearly shows that the HPF planner is computationally more efficient than the FM method as N increases.
3) The following section explains the performance improvement in real-time computation time with the new structure of the HPF planner and motion controller (goal-seeking approach) with respect to the FM method and motion controller (path-tracking approach). In the FM approach, the complete path coordinates are precalculated before the UGV movement starts. The predetermined reference path has N_p discrete points, i.e., $\{(x_0, y_0), (x_1, y_1), \ldots, (x_{N_p}, y_{N_p})\}$, from the source to the goal. At each sampling instant t_i, the current position of the UGV (x_0, y_0) is determined using template matching. Each of the N_p points on the predetermined path is checked to

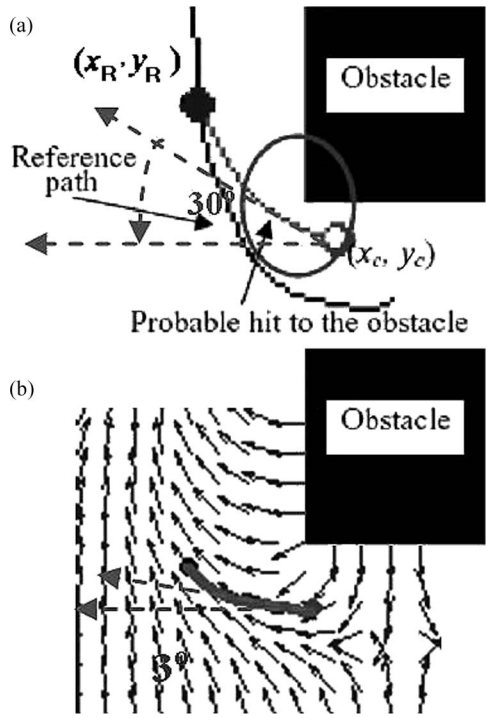


Fig. 17. Behavioral comparison between path-tracking and goal-seeking problems.

find out which point is closest to (x_0, y_0). The reference point (x_R, y_R) on the path is determined such that it is at a set distance (d_0) from the closest point. Thus, at each t_i, it will take N_p number of real-time checks on the path to find out the closest point and the corresponding reference position. When HPF is combined with the quadratic controller, at each t_i, the reference point computation in real time will take δL additions. This decreases the real-time computation cost and time by a factor of $\lambda_p = N_p/\delta L$, where N_p is the total number of points from source to destination on the path and δL is the discrete look-ahead distance toward the destination and where the following will always be satisfied:

$$N_p \geq \delta L, \qquad \therefore\ \lambda_p \geq 1. \tag{13}$$

The longer the path (N_p) is, the more efficient the HPF planner will be.

VI. Experimental Testing of Suggested NCS

The capabilities and features of the suggested system are demonstrated using five case studies. In the case of the HPF planner, the reference path from source to destination is calculated and compared with the actual path of the UGV. The distance between the UGV's position and the ideal path is calculated at each time t_i with respect to the time delay. This difference is called the distance error. It shows how close the actual path is to the ideal path. The total time to reach the destination is also recorded. The distance error and the total time are the main criteria used for performance evaluation.

A. Case 1: Different Obstacle and Background Cases

Fig. 18 shows the satisfactory performance of the suggested method with different numbers, shapes, and sizes of obstacles in the workspace. We observe that all the turns around the obstacle are at enough safe distance. Without tracking the ideal path, the reference array approach yields the average and the maximum distance error of 2–5 cm, respectively, which is 1% error with respect to the dimension of the space.

B. Case 2: Network and Processing Delays

Generally speaking, the network delay is on the order of a few hundreds of milliseconds on the Internet. Typical delays observed in the U.S. continent are 0.1–0.6 s. We, however, tested the functionality for a network delay range of 0.1–1.2 s. Fig. 19 shows that the distance error increases exponentially with an increase in network delay. For the average network delay of 0.3 s, we have a mean distance error of 1 cm and a maximum error of less than 5 cm. Even with delays as large as 1.2 s, the maximum error is 20 cm (4% of the diagonal length of workspace area).

C. Case 3: Path Quality

Figs. 20 and 21 show that the path from the FM-based NC exhibits more variations in the curvature graph (sudden turns) unlike HPF which provides a smoother path. In other words, the path obtained from the suggested NC (HPF-based approach) is more dynamically friendly to traverse than the path obtained from the FM-based approach. Figs. 22 and 23 also clearly show that the path generated by the suggested approach is less susceptible to delay-induced errors than the one from the FM-based approach. It can be clearly seen that a goal-seeking mode significantly improves performance.

D. Case 4: Special Cases

An interesting intelligent behavior is observed during the experiments when the UGV is initially oriented in an opposite direction to the gradient guidance field. The UGV simultaneously combines the basic behaviors: driving in reverse, turning around, staying away from the obstacle, and proceeding toward the target to synthesize a human-driver-like maneuver that treats the space as a scarce resource (Fig. 24).

HPF displays an intelligent behavior when the goal point chosen is unreachable due to a barrier in the workspace. We deliberately put an obstacle so that the workspace is divided into two separate regions as shown in Fig. 25. We kept the UGV in one region and chose the destination point in another region that is unreachable by the UGV due the wall-like obstacle dividing the workspace into isolated regions. The complete region around the UGV is pulled down at an equal high potential. This created a zero gradient (flat region) around the UGV. V_x and V_y are both zero, giving the next reference point for the UGV to be the same point as per (10). Therefore, the UGV does not move from its position as if knowing beforehand that the goal point is unreachable (Fig. 25).

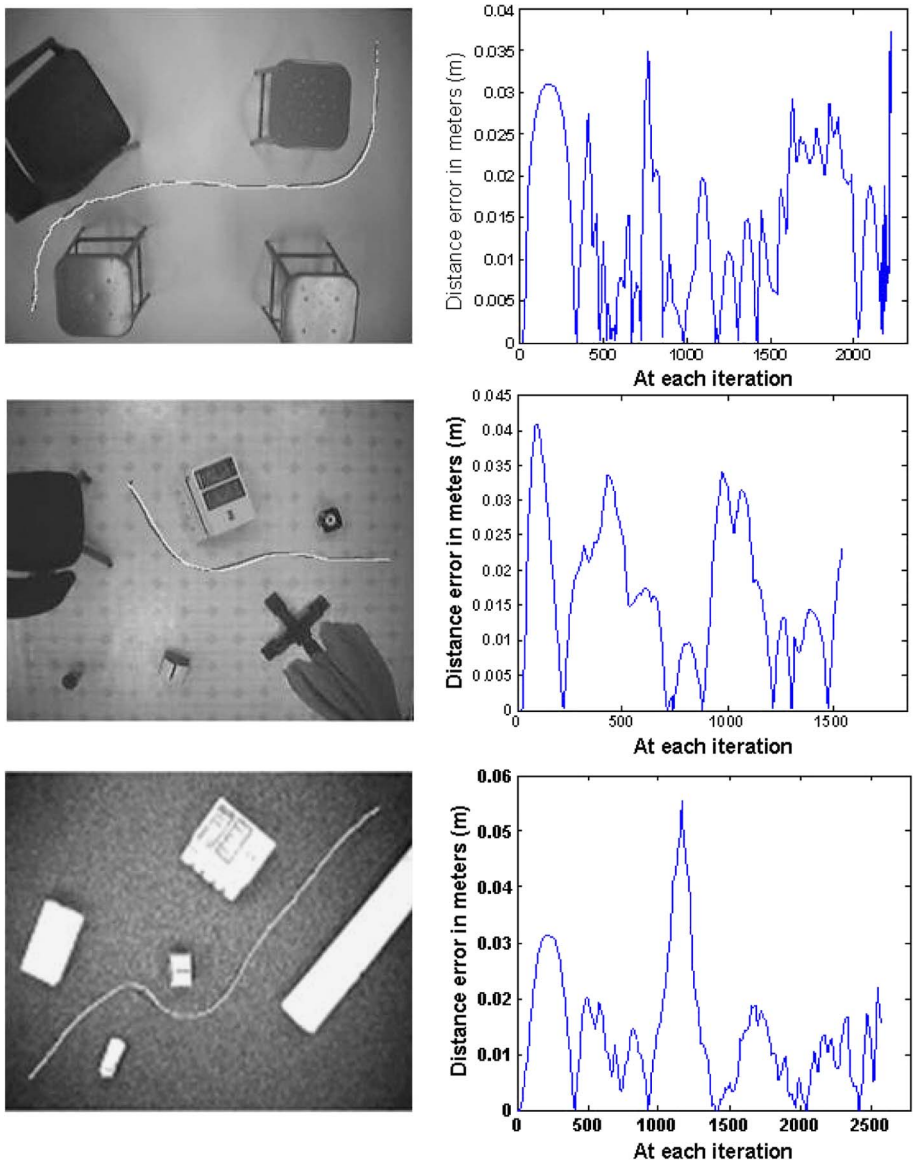


Fig. 18. Workspace images and the graph of distance error versus time.

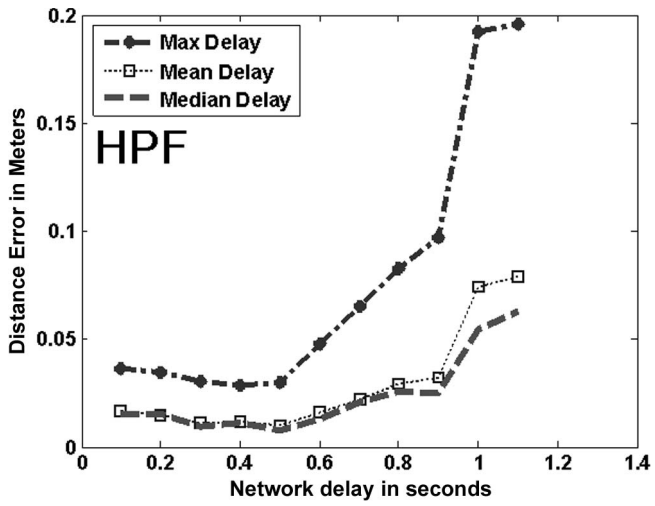


Fig. 19. Distance error measures versus network delay.

E. Case 5: Robustness to Lighting Conditions

Reliable edge detection is the backbone of the vision module. The harmonic potential planner can use a raw edge map to generate the guidance field without any need to interpret the map's contents (i.e., the contents of the map are unknown to the operator). There are several difficulties that could arise when the edges of a nonsynthetic image are to be detected. Noise, shadows, variable texture, or contrasts in the image could all cause false edge components. We performed a few experiments for different lighting and contrast conditions for one image. In Fig. 26, we observe that we loose edges as well as get extra edges as the lighting condition changes. However, the HPF planner successfully generates paths from source to destination in all the cases (black line showing the generated path). The fact that the NC did not show much sensitivity to missing edges is not luck. It has to do with the choice of the planning module. Harmonic potential is intimately connected

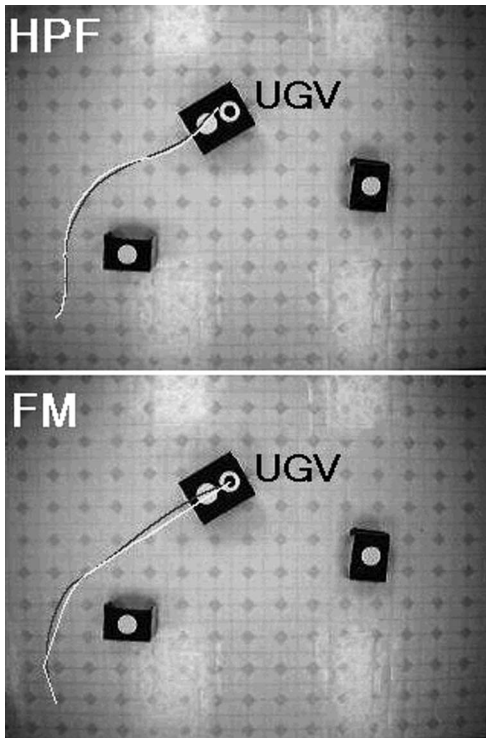


Fig. 20. HPF planner versus FM planner.

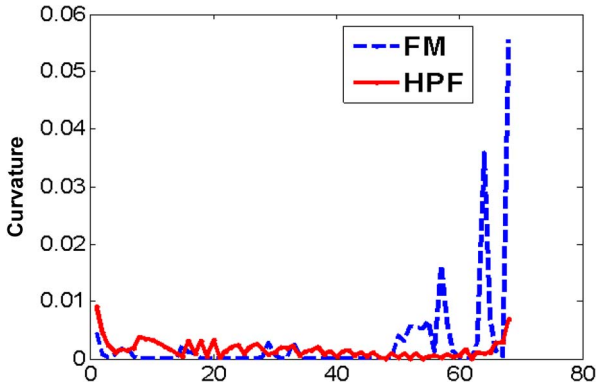


Fig. 21. Curvatures for Fig. 20.

to collision probability [40]. The HPF planner guides motion to reduce this probability. Therefore, if part of the edge contour is missing, the probability of collision at the missing part (i.e., the harmonic potential) is still high, and the planner will attempt to guide motion away from it. These are just preliminary results to demonstrate that the HPF planner has a considerable level of tolerance to artifacts in the edge map or to the loss of authentic edges and the appearance of false ones.

F. Note on Multi-UGV Case

While the focus of this paper is on networking the modules for a single UGV, the suggested controller has the ability to operate in a multi-UGV environment. This ability stems from fundamental capabilities which the HPF approach has. The HPF planner is naturally decentralized, making it a valid guidance protocol in a multiagent environment where each agent uses the HPF approach for guidance. This means that NCs similar to the one suggested in this paper can act independently

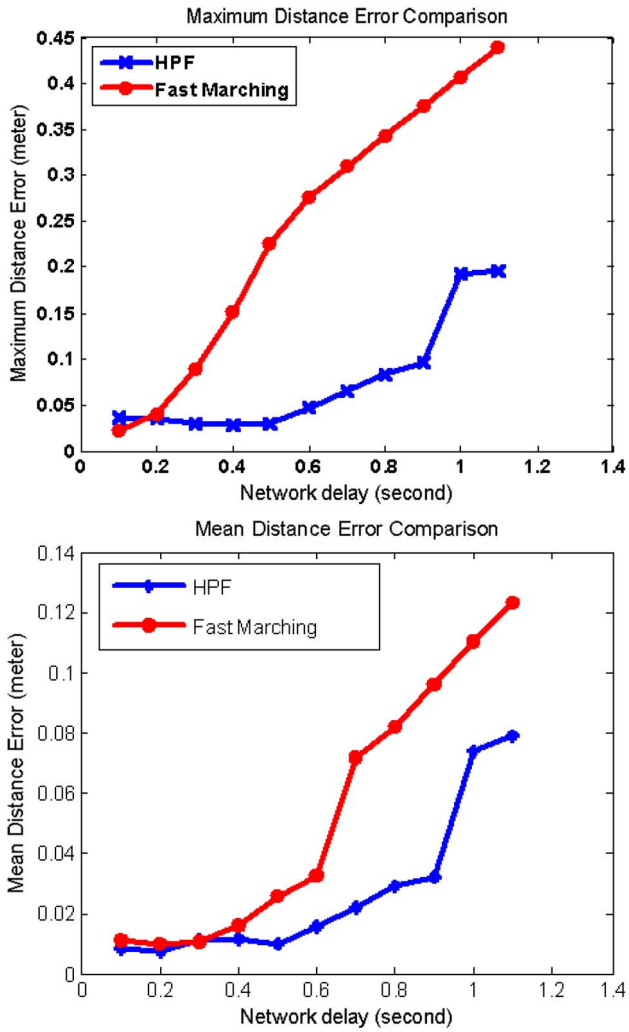


Fig. 22. Maximum and mean distance error versus network delay comparing FM and HPF planner.

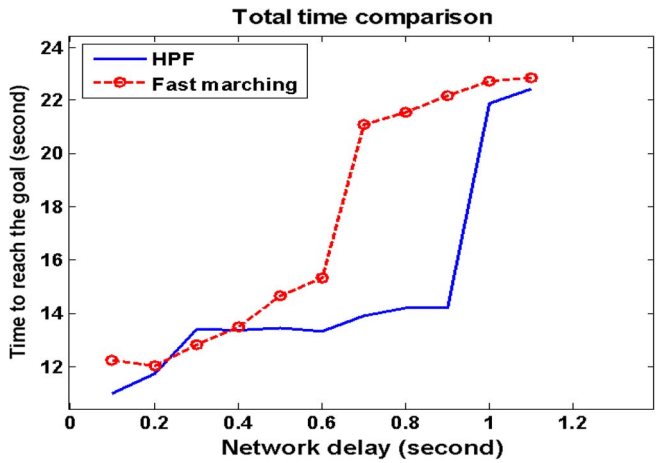


Fig. 23. Total time versus network delay comparing FM and HPF.

on robots sharing the same workspace yet collectively enable each UGV to safely proceed to its target (Fig. 27).

Despite the fact that the controllers are not aware of each others' presence, the overall control action regulating the group of robots in the workspace is a distributed evolutionary self-organizing one.

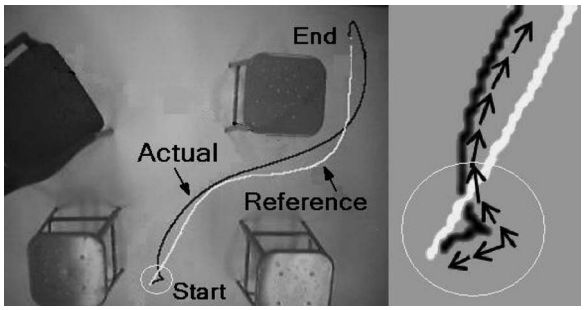


Fig. 24. Direction reversal maneuver. Arrows show the UGV orientation.

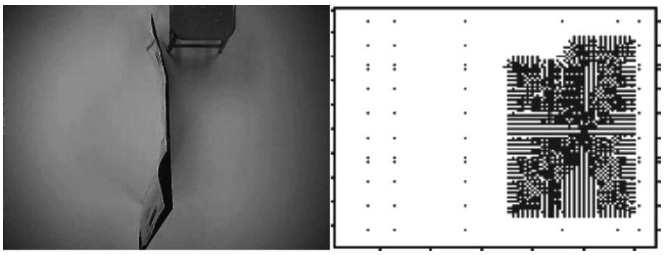


Fig. 25. Image with a barrier separating two regions.

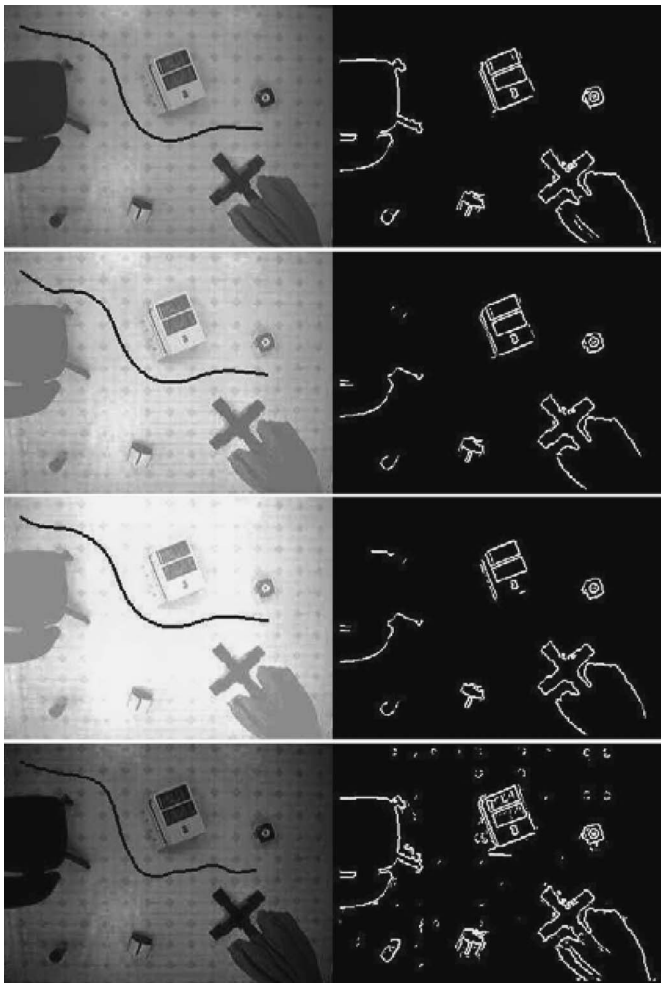


Fig. 26. Experiments with different lighting conditions.

Fig. 28 shows five robots controlled individually by an HPF planner. Each robot is required to move from a start point (S) to a target point (T). As can be seen, the decentralized collective

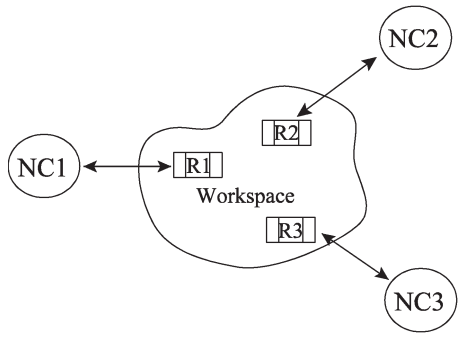


Fig. 27. Suggested NC is naturally decentralized.

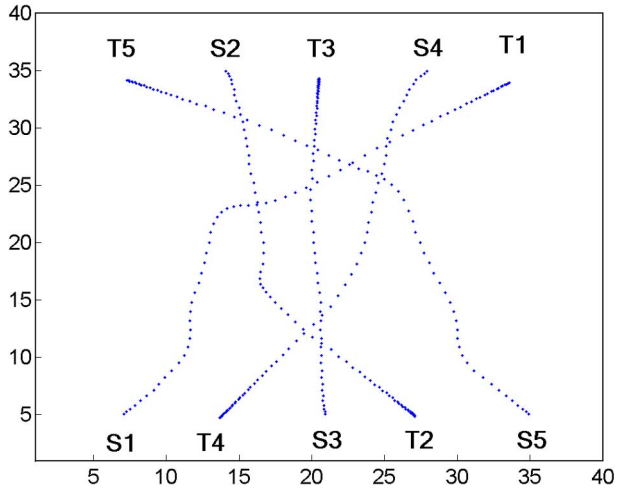


Fig. 28. HPF planners have full awareness of the other agents in the environment.

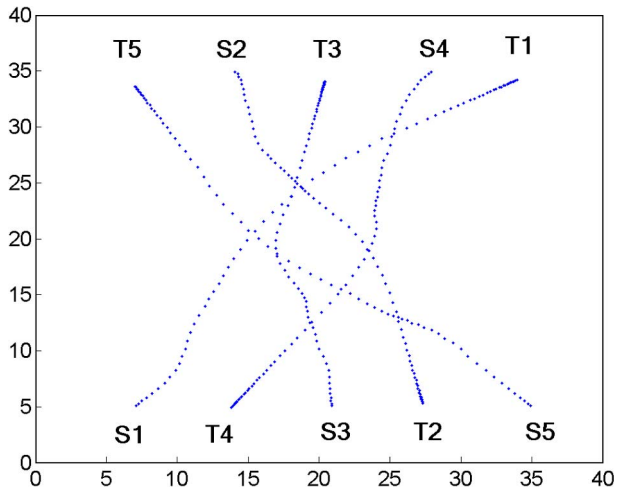


Fig. 29. Each HPF planner is only aware of its nearest neighbor.

of harmonic planners successfully generates a safe trajectory for each robot to its target. In Fig. 29, a close to a catastrophic failure situation is created. The individual planners neglect the presence of all the robots in the workspace except the one closest to the robot they are steering. The simulation results are shown in Fig. 29. The group kept functioning normally as if nothing happened.

VII. Conclusion, Discussion, and Future Work

This paper has successfully shown through experimental results that edge detection, an HPF planner, and a network-based quadratic controller can be networked to create an efficient and delay-tolerant integrated NC. The edge-detection and HPF-based planning modules satisfy several important features that are central to the integration of the system. This gives more generality and flexibility to the UGV environment which is not the case with template matching. The efficiency of the suggested approach is evident from the comparison with the FM method. Moreover, the memory storage needed by the method is negligible (on the order of kilobytes), and the computations can be carried out in place.

We observe that the ratio of the number of edge pixels (information pixels) to the total number of pixels in the image is always less than one. In case of iSpace, we have observed it to be on the average 5% of the whole image pixels. Current results for the NC are obtained for whole frame transmission. If an IP camera or hardware with local processing is used to transmit only the edge pixels, a much superior performance compared with the one we already have is expected. The NC will use lesser bandwidth than required for the whole frame transmission. As for scalability, the following can be observed.

1) All the components of the NCS can be built/implemented using cheap easily available off-the-shelf hardware components. Thus, adding more cameras, UGVs, or NCs with different network addresses to the workspace is a very quick and easy job, ensuring the scalability.
2) It is possible to add any number of cameras in the ceiling to increase the workspace area coverage. As the tilt angle is zero and the cameras are centered in the workspace, they can be spaced in the ceiling in such a way that they form an array of cameras to cover a larger workspace. The edge maps from the different cameras can be easily combined using a logic OR operation.
3) Currently, the NC is programmed to handle only one agent (UGV); however, HPF can be naturally extended to control multirobots with very little modification to the existing structure as explained in [41]. Each agent can be assigned one NC, and all other agents can be treated as moving obstacles.

There is still a lot of work that needs to be done in order to extend the capabilities of the suggested approach. For example, the robot end effector is expected to carry a payload. While considering the kinematic aspects only at the high-level control stage is sufficient for many practical cases, accommodating dynamics at the high-level control stage will have to be addressed.

References

[1] G. McKee, "What is networked robotics," in *Informatics in Control Automation and Robotics*, vol. 15, *Lecture Notes in Electrical Engineering*, J. Andrade-Cetto, J. Ferrier, J. Pereira, and J. Filipe, Eds. Berlin, Germany: Springer-Verlag, 2008, p. 35.

[2] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proc. IEEE ICRA*, 1985, pp. 500–505.

[3] C. I. Connolly, J. B. Burns, and R. Weiss, "Path planning using Laplace's equation," in *Proc. IEEE ICRA*, 1990, pp. 2102–2106.

[4] K. Sato, "Collision avoidance in multi-dimensional space using Laplace potential," in *Proc. 15th Conf. Robot. Soc. Jpn.*, 1987, pp. 155–156.

[5] D. Alvarez, J. C. Alvarez, and R. C. Gonzalez, "Online motion planning using Laplace potential fields," in *Proc. IEEE ICRA*, 2003, pp. 3347–3352.

[6] J.-O. Kim and P. K. Khosla, "Real-time obstacle avoidance using harmonic potential functions," *IEEE Trans. Robot. Autom.*, vol. 8, no. 3, pp. 338–349, Jun. 1992.

[7] S. Masoud and A. Masoud, "Motion planning in the presence of directional and obstacle avoidance constraints using nonlinear anisotropic, harmonic potential fields: A physical metaphor," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 32, no. 6, pp. 705–723, Nov. 2002.

[8] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.

[9] B. Siciliano and O. Khatib, *Handbook of Robotics*. Berlin, Germany: Springer-Verlag, 2008.

[10] A. A Masoud, "An informationally-open, organizationally-closed control structure for navigating a robot in an unknown, stationary environment," in *Proc. IEEE Int. Symp. Intell. Control*, 2003, pp. 614–619.

[11] S. Masoud and A. Masoud, "Constrained motion control using vector potential fields," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 30, no. 3, pp. 251–272, May 2000.

[12] A. A. Masoud, "Managing the dynamics of a harmonic potential field-guided robot in a cluttered environment," *IEEE Trans. Ind. Electron.*, vol. 56, no. 2, pp. 488–496, Feb. 2009.

[13] S. M. Amin, E. Y. Rodin, M. K. Meusey, T. W. Cusick, and A. Garcia-Ortiz, "Evasive adaptive navigation and control against multiple pursuers," in *Proc. ACC*, 1997, pp. 1453–1457.

[14] V. I. Utkin, S. V. Drakunov, H. Hashimoto, and F. Harashima, "Robot path obstacle avoidance control via sliding mode approach," in *Proc. IEEE/RSJ Int. Workshop Intell. Robots Syst., Intell. Mech. Syst.*, 1991, pp. 1287–1290.

[15] L. C. A. Pimenta, A. R. Fonseca, G. A. S. Pereira, R. C. Mesquita, E. J. Silva, W. M. Caminhas, and M. F. M. Campos, "On computing complex navigation function," in *Proc. IEEE ICRA*, 2005, pp. 3452–3457.

[16] R. Kelly, V. Sanchez, E. Bugarin, and H. Rodriguez, "A fixed-camera controller for visual guidance of mobile robot via velocity fields," in *Proc. IEEE ICRA*, 2005, pp. 3137–3142.

[17] W. Snyder and H. Qi, *Machine Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[18] A. A. Masoud and M. Bayoumi, "Using local structure for the reliable removal of noise from the output of the LoG edge detector," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 2, pp. 328–337, Feb. 1995.

[19] F. van der Heijden, "Edge and line feature extraction based on covariance models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 1, pp. 16–33, Jan. 1995.

[20] K. Yoshizawa, H. Hashimoto, M. Wada, and S. M. Mori, "Path tracking control of mobile robots using a quadratic curve," in *Proc. IEEE Intell. Veh. Symp.*, Tokyo, Japan, 1996, pp. 58–63.

[21] M.-Y. Chow, "Guest editorial on the special section on distributed network-based control systems and applications," *IEEE Trans. Ind. Electron.*, vol. 51, no. 6, p. 1126, Dec. 2004.

[22] Y. Tipsuwan and M.-Y. Chow, "Gain scheduler middleware: A methodology to enable existing controllers for networked control and teleoperation—Part I: Networked control," *IEEE Trans. Ind. Electron.*, vol. 51, no. 6, pp. 1218–1227, Dec. 2004.

[23] Y. Tipsuwan and M.-Y. Chow, "Gain scheduler middleware: A methodology to enable existing controllers for networked control and teleoperation—Part II: Teleoperations," *IEEE Trans. Ind. Electron.*, vol. 51, no. 6, pp. 1228–1237, Dec. 2004.

[24] W.-L. D. Leung, R. Vanijjirattikhan, Z. Li, L. Xu, T. Richards, B. Ayhan, and M.-Y. Chow, "Intelligent space with time sensitive applications," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics*, 2005, pp. 1413–1418.

[25] M. Khosravi and R. W. Schafer, "Template matching based on a grayscale hit-or-miss transform," *IEEE Trans. Image Process.*, vol. 5, no. 6, pp. 1060–1066, Jun. 1996.

[26] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts," *Proc. Nat. Acad. Sci. U.S.A. (PNAS)*, vol. 93, no. 4, pp. 1591–1595, Feb. 1996.

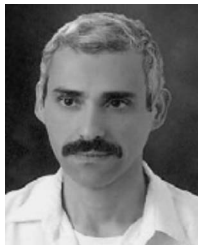
[27] C. Petres, Y. Pailhas, J. Evans, Y. Petillot, and D. Lane, "Underwater path planning using fast marching algorithms," in *Proc. Oceans Eur.*, 2005, pp. 814–819.

[28] R. Gonzalez and R. Woods, *Digital Image Processing*. Upper Saddle River, NJ: Pearson, 2004.
[29] G. Baliga and P. R. Kumar, "A middleware for control over networks," in *Proc. IEEE Conf. Decisions Controls*, 2005, pp. 482–487.
[30] Z. Qiuming, "Hidden Markov model for dynamic obstacle avoidance of mobile robot navigation," *IEEE Trans. Robot. Autom.*, vol. 7, no. 3, pp. 390–397, Jun. 1991.
[31] P. Arena, L. Fortuna, M. Frasca, G. Lo Turco, L. Patane, and R. Russo, "Perception-based navigation through weak chaos control," in *Proc. IEEE Conf. Decisions Controls*, 2005, pp. 221–226.
[32] P. I. Corkey and P. Ridley, "Steering kinematics for a center-articulated mobile robot," *IEEE Trans. Robot. Autom.*, vol. 17, no. 2, pp. 215–218, Apr. 2001.
[33] L. Litz, O. Gabel, and I. Solihin, "NCS-controllers for ambient intelligence networks—Control performance versus control effort," in *Proc. IEEE Conf. Decisions Controls*, 2005, pp. 1571–1576.
[34] J. Yen and N. Pfluger, "A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 6, pp. 971–978, Jun. 1995.
[35] A. D. Tews, G. S. Sukhatme, and M. J. Mataric, "A multi-robot approach to stealthy navigation in the presence of an observer," in *Proc. IEEE ICRA*, Apr. 26–May 1, 2004, vol. 3, pp. 2379–2385.
[36] G. N. Desouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 2, pp. 237–267, Feb. 2002.
[37] J. Tisdale, A. Ryan, M. Zennaro, X. Xiao, D. Caveney, S. Rathinam, J. K. Hedrick, and R. Sengupta, "The software architecture of the Berkeley UAV platform," in *Proc. IEEE Int. Conf. Control Appl.*, 2006, pp. 1420–1425.
[38] MIT AI Lab Memo No. 1293 R. A. Brooks, Intelligence Without Reason, Apr. 1991.
[39] T. Rofer and M. Jungel, "Vision-based fast and reactive Monte-Carlo localization," in *Proc. IEEE ICRA*, 2003, pp. 856–861.
[40] C. I Connolly, "Harmonic functions and collision probabilities," *Int. J. Robot. Res.*, vol. 16, no. 4, pp. 497–507, Aug. 1997.
[41] A. Masoud, "Decentralized, self-organizing, potential field-based control for individually-motivated, mobile agents in a cluttered environment: A vector-harmonic potential field approach," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 37, no. 3, pp. 372–390, May 2007.
[42] S. Todorovic and M. C. Nechyba, "A vision system for intelligent mission profiles of micro air vehicles," *IEEE Trans. Veh. Technol.*, vol. 53, no. 6, pp. 1713–1725, Nov. 2004.
[43] K. Althofer, D. Fraser, and G. Bugmann, "Rapid path planning for robotic manipulators using an emulated resistive grid," *Electron. Lett.*, vol. 31, no. 22, pp. 1960–1961, Oct. 26, 1995.
[44] M. Stan, W. Burleson, C. Connolly, and R. Grupen, "Analog VLSI for robot path planning," *Analog Integr. Circuits Signal Process.*, vol. 6, no. 1, pp. 61–73, Jul. 1994.
[45] B. Girau and A. Boumaza, "Embedded harmonic control for dynamic trajectory planning on FPGA," in *Proc. 25th IASTED Int. Multi-Conf., Artif. Intell. Appl.*, Innsbruck, Austria, 2007, pp. 244–249.
[46] P. Vadakkepat, T. Lee, and L. Xin, "Application of evolutionary artificial potential field in robot soccer system," in *Proc. Joint 9th IFSA World Congr., 20th NAFIPS Int. Conf.*, Vancouver, BC, Canada, Jul. 25–28, 2001, vol. 5, pp. 2781–2785.
[47] K. Sim, K. Byun, and F. Harashima, "Internet-based teleoperation of an intelligent robot with optimal two-layer fuzzy controller," *IEEE Trans. Ind. Electron.*, vol. 53, no. 4, pp. 1362–1372, Aug. 2006.
[48] D. Nguyen, S. Oh, and B. You, "A framework for Internet-based interaction of humans, robots, and responsive environments using agent technology," *IEEE Trans. Ind. Electron.*, vol. 52, no. 6, pp. 1521–1529, Dec. 2005.

Rachana Ashok Gupta received the B.Eng. degree in electronics and telecommunication from Maharashtra Institute of Technology, Pune, India, in 2002 and the M.Sc. degree in electrical engineering from North Carolina State University (NCSU), Raleigh, in 2006, where she is currently working toward the Ph.D. degree in electrical engineering in the Department of Electrical and Computer Engineering.

From 2002 to 2004, she was a Software Engineer in i-Flex Solutions Ltd., Mumbai, India. She has been a part of the Advanced Diagnosis, Automation, and Control Laboratory, NCSU, since January 2005, working on robotics, vision, and control area.

Ms. Gupta has been a member of the Honor Society of Phi Kappa Phi since March 2006.



Ahmad A. Masoud (M'83) received the B.Sc. degree in electrical engineering with a major in power systems and a minor in communication systems from Yarmouk University, Irbid, Jordan, in 1985 and the M.Sc. (digital signal processing) and Ph.D. (robotics and autonomous systems) degrees in electrical engineering from Queen's University, Kingston, ON, Canada, in 1989 and 1995, respectively.

From 1985 to 1987, he was a Researcher with the Department of Electrical Engineering, Jordan University of Science and Technology, Irbid. He was also a part-time Assistant Professor and a Research Fellow with the Department of Electrical and Computer Engineering, Royal Military College of Canada, Kingston, from 1996 to 1998. During that time, he carried out research in digital signal processing (DSP)-based demodulator design for high-density multiuser satellite systems and taught courses in robotics and control systems. He is currently an Assistant Professor with the Electrical Engineering Department, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia. His current interests include navigation and motion planning, robotics, constrained motion control, intelligent control, and DSP applications in machine vision and communication.

Mo-Yuen Chow (S'81–M'82–SM'93–F'07) received the B.S. degree in electrical and computer engineering from the University of Wisconsin, Madison, in 1982 and the M.Eng. and Ph.D. degrees from Cornell University, Ithaca, NY, in 1983 and 1987, respectively.

In 1989, he joined the Department of Electrical and Computer Engineering, North Carolina State University (NCSU), Raleigh, as an Assistant Professor, where he became an Associate Professor in 1993 and has been a Professor since 1999. He has established the Advanced Diagnosis, Automation, and Control Laboratory, NCSU. His research focuses on fault diagnosis and prognosis, distributed control, and computational intelligence. He has been applying his research to areas including mechatronics, power distribution systems, distributed generation, motors, and robotics. He has published one book, several book chapters, and over 100 journal and conference articles related to his research work.

Dr. Chow was the recipient of the IEEE Region-3 Joseph M. Biedenbach Outstanding Engineering Educator Award and the IEEE Eastern North Carolina Section Outstanding Engineering Educator Award. He is a Coeditor-in-Chief for the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS and an Associate Editor for the IEEE TRANSACTIONS ON MECHATRONICS.