Proceedings of the 45th IEEE Conference on Decision & Control
Manchester Grand Hyatt Hotel
San Diego, CA, USA, December 13-15, 2006



# A Network based, Delay-tolerant, Integrated Navigation System for a differential drive UGV using Harmonic Potential Field

Rachana Ashok Gupta*, Ahmad A Masoud**, Mo-Yuen Chow*
ragupta@unity.ncsu.edu, masoud@kfupm.edu.sa, chow@eos.ncsu.edu
*Electrical and Computer Engineering Department, North Carolina State University, Raleigh, NC – 27606, USA
**Electrical Engineering Department, KFUPM, P.O.Box 287, Dhahran 31261, Saudi Arabia

***Abstract*- Network based integrated navigation systems are a multidisciplinary effort whose aim is to produce a network structure and components that are capable of integrating sensors, actuators, communication, and control algorithms in a manner to suit real-time navigation and/or obstacle avoidance. There are many challenges to be overcome in order to put such a distributed and heterogeneous system together. This paper deals with some of these issues, i.e. the adverse effect of processing delays on the system and efficient integration of different modules. This paper describes a delay-resistant sensory-motor module for navigating a differential drive unmanned ground vehicle (UGV) in an indoor cluttered environment. The module consists of an early vision edge detection stage, a harmonic potential field (HPF) planner; a network based quadratic curve fitting controller and gain schedule middleware, (GSM). Though the different techniques and algorithms used to implement the navigation system have been previously well-studied algorithms, the novel contribution in this paper is the way all these different modules are integrated together for the first time to create an efficient structure for a network based integrated navigation system making it easy to implement and realize for many different applications. The structure of this module and its components are described. Thorough experimental results along with performance assessment comparing to the previous implementation are also provided to illustrate the effectiveness of the new integrated navigation structure.**

## I. INTRODUCTION

A Network based Integrated navigation system (NBINS) has different modules combined together to guide a UGV (Unmanned Ground Vehicle) from one point to another in the space of interest, where the navigation intelligence lies on a main controller away from the UGV[14]. Advantages of such a multidisciplinary system are many. Network controllers allow data to be efficiently shared making the system scalable and the solutions globally optimal. They eliminate unnecessary wiring, and most importantly, they connect cyber space to physical space making task execution remotely accessible (a form of tele-presence).

We developed a closed-loop NBINS called Intelligent space (iSpace) consisting of different modules like image processing, automation control, communication and networking for navigation of a differential drive UGV in an indoor 2-D environment (**Fig. 1**) iSpace was used as a testbed in order to validate our method experimentally as well as to compare our method with other popular techniques which use template matching for recognition and fast marching for path planning. Therefore, iSpace was initially implemented using three basic modules: (i) a template matching-based vision system to recognize the contents of the environment, the UGV and their configurations [12], (ii) a planning module for computing the optimal, obstacle-free, reference path of the UGV using the fast marching method [13] and (iii) a tracking module that uses the sequence of control commands generated by the network controller in the closed-loop navigational system.

There are several challenges facing the reliable operation of the above system. One of the most important issues is that of network and processing delays and interface between different modules. Delays can destabilize the closed loop, visual, servo system. Therefore, they have to be compensated or kept to a minimum. They are caused by several factors such as: image flow from the camera to the controller, data processing, and network congestion. The use of template matching for workspace content recognition suffers from a flexibility issue in the sense that the shape of each component of the environment has to belong to an *a priori* known set of shapes used as templates. Also the use of the fast marching method to generate the reference path imposes limitations on the operation of the system. The generated path has to be tracked by the system regardless of how far off the delays cause the UGV to deviate from the reference in the due course of movement. Consequently, the result may not be always an intelligent decision considering path and time optimality.[14]

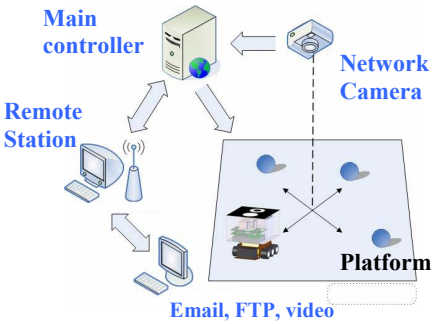



Fig. 1. iSpace configuration and flow diagram as a network based integrated mechatronics system.

We emphasize on lowering the processing delay part of the whole system. A computationally efficient, generic and delay-tolerant structure is described in this paper. Edge detection is used to acquire the obstacle map of the workspace. This makes it possible to capture a full representation of the environment without making restrictive assumptions regarding its contents. Edge detection is also more computationally efficient than template matching. The HPF approach is used for motion planning [2][8]. The gradient directional output of HPF converts the path tracking problem into a goal seeking problem. Therefore the UGV can be driven to the goal from any point in the workspace. This causes a reduction in the real time computational delay making it compatible with the network based controller. The HPF-based planner is a single module in an interconnected structure,

simultaneously satisfying several important features that are central to a NBINS. Those features and ones possessed by the other modules are discussed later in the paper.

## II. PATH PLANNING AND PATH TRACKING FOR NBINS

The NBINS considered in this paper (iSpace) has the following components (as shown in Fig. 1): (i) Overhead web camera as a global sensor, (ii) a differential drive UGV used as the navigator with no on-board sensing capabilities, and (iii) a network controller with user interface, that can be located on any remote computing interface in the world.

Network delay is more random in nature than the processing time delay. Therefore stochastic and predictive methods are used to deal with the network delay. Processing delay may be reduced by adopting faster and computationally more efficient algorithms on the main controller. Our paper deals with the processing delay part of this network based integrated system. Our approach also emphasizes compatibility between the different modules of the system. This compatibility makes it possible for a module to directly accept the output from another module without any conditioning required, hence further significantly reducing the processing delay and the chance of inter-module conflict. While this is one criterion guiding our selection of the vision, planning, and actuation stages, another criterion is also considered that has to do with the flexibility each module exhibits in terms of avoiding restrictive assumptions on the type of data it is required to process. We believe that these criteria are the key to building a system that will behave in the desired manner.

### *A. Edge Detection and model based 2-D HPF*

Edge detection is a classic early vision tool used to extract discontinuities from an image. The discontinuities provide a skeleton representation of the UGV's environment that marks the boundaries of all the entities it contains. The edge detector used in this paper has two branches. One branch process the image with a circularly symmetric Laplacian of Gaussian operator (2). [14]

$$G(\sigma, x) = \exp\left(x^2 + y^2 / 2\sigma^2\right), \nabla G(x, y) = \left[\frac{\partial G}{\partial x} \quad \frac{\partial G}{\partial y}\right]^T \quad (1), (2)$$

$$\nabla^2 G(x,y) = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2} \quad (3)$$

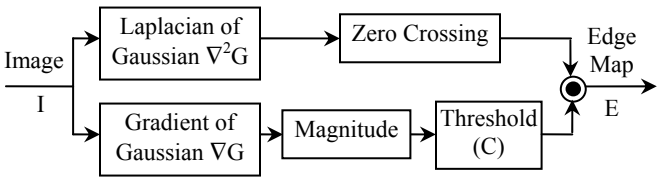



Fig. 2. Complete edge detector block diagram

The zero crossing contours from this operator are taken as the candidate edges. The edge filtering adopted here is based on the strength of the edge, i.e. its contrast. Edges above a certain threshold are accepted as authentic; otherwise they are rejected as noise. An approximated Gaussian first derivative gradient operator (1) is used to produce an estimate of the contrasts. The 2-D Gaussian function is shown in (3). [14]. Fig. 2 shows the complete block diagram of the edge detector.

Fig. 3 shows typical indoor workspace images and their edge maps. In the current settings, as we consider indoor environment for UGV navigation, we assume that it is possible to maintain enough illumination from the top such that no important part of the workspace, especially with the obstacles, is in the dark. Obstacles are assumed to be stationary. These constraints will be relaxed in future research.

We can observe that edge detection offers flexibility with any shape, size and number of obstacles in an indoor workspace removing the need for a dedicated template for each obstacle. Fig. 4 shows that template matching marks only the obstacles with templates and on the other hand, edge detection marks the definite boundaries of all the obstacles for the same workspace. Edge detection output is expressed in Eq(4).

$$E(x_i, y_j) = 1 \text{ if } (x_i, y_j) \in \Gamma \quad (4)$$
$$= 0 \text{ if } (x_i, y_j) \notin \Gamma \qquad \text{for all } (i, j)$$

Where $E(x, y)$ is the image representing the edge map and $\Gamma$ is the set of edge/obstacle boundary points for all obstacles in workspace.

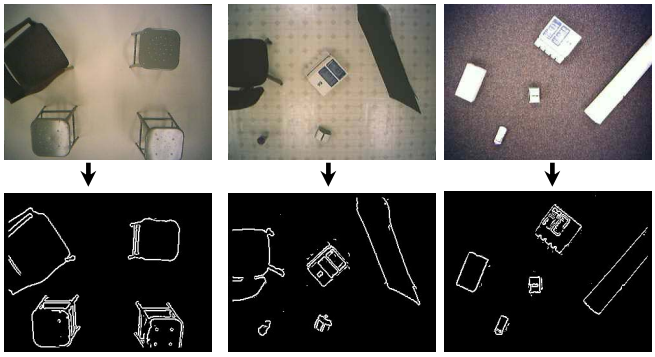


**Fig. 3**. Edge detection results for different obstacle shapes, sizes and different background.

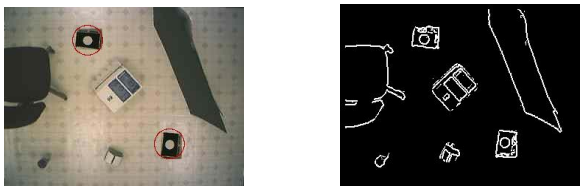


**Fig. 4**.. Template matching and Edge detection outputs for the same workspace image

The HPF method is used for path planning. [14] The use of potential field in motion planning was introduced by Khatib [1]. The principle behind this was - the obstacles were represented by the repelling force and the point of destination was represented by the attractive force. The local minima problem in Khatib's method was removed by the use of the HPF approach [2]. In HPF the navigation potential ($\phi$) is obtained by solving the Laplace equation as expressed in (5).

$$\nabla^2 \phi(x, y) \equiv 0 \quad x, y \in \Omega \quad (5)$$
$$\text{subject to } \phi(x, y) = 1 \quad x, y \in \Gamma$$
$$\text{and} \quad \phi(x, y) = 0 \quad (x, y) = (x_T, y_T)$$

Where $\nabla^2$ is the Laplace operator, $\Omega$ is the workspace of the UGV ($\Omega \subset \Re^2$), $\Gamma$ is the boundary of the obstacles, and $(x_T, y_T)$ is the target point. The above setting is known as the Dirichlet's setting and is used in this paper for generating the navigation HPF. The obstacle\ free path to the target is generated by traversing the negative gradient of $(\phi)$ i.e. $(-\nabla\phi)$. As it has been emphasized earlier in [14], the close relation between edge detection and HPF is what makes this path planning algorithm so interestingly efficient and flexible. As with edge detection the obstacle boundaries are already raised to a high potential (binary edges, $\phi = 1$) and all the other points in the workspace, which are not on the obstacle contour, are at a low equal potential (robot movement area $\phi = 0$); (compare Eq(4) and Eq(5)) thus it provides the exact raw data required for HPF in the Dirichlet's setting to create the gradient direction matrix. Numerical solutions for the Laplace's equation are readily obtained from the finite difference method. As elaborated in [2], this method simply consists of repeatedly replacing each grid points with the average of its neighbors using successive relaxation until the array $u$ contains a sampling $\phi$ of where every non-boundary condition node has a neighbor with a smaller value representing a negative gradient except the goal point. All the obstacles with edges as closed contours will be represented by a constant high potential (Fig. 5). The normalized gradient at each point represents the directional guidance at that point in the workspace as represented by the blue arrows in Fig. 5.

$$V_x = \frac{-\partial\phi/\partial x}{\sqrt{\left(\partial\phi/\partial x\right)^2 + \left(\partial\phi/\partial y\right)^2}}, \quad V_y = \frac{-\partial\phi/\partial y}{\sqrt{\left(\partial\phi/\partial x\right)^2 + \left(\partial\phi/\partial y\right)^2}} \qquad (6)$$

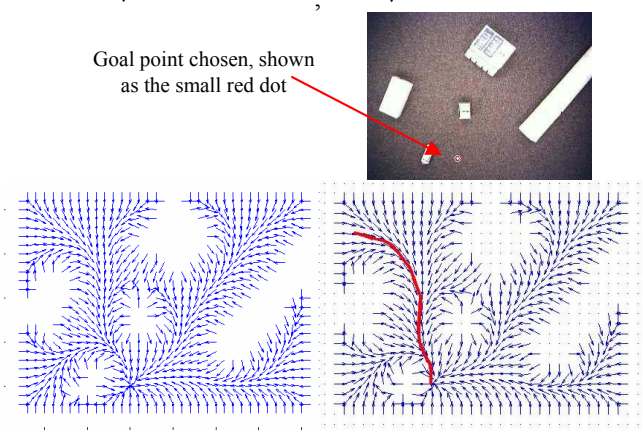



Fig. 5(a) and (b) Region-to-point feature of HPF gradient.

It is evident that all the points in the workspace steer the UGV away from obstacles and to the goal highlighting the HPF planner's *region-to-point* guidance property. Thus, the UGV can follow the negative gradient at each point from source to reach the destination. Fig. 5(b) displays the path from one of the workspace points to the chosen destination.

*B. HPF, a region to point guidance function, combined with the network based controller*

The Quadratic curve fitting path tracking controller is used for the movement control of the UGV. The basic principle of quadratic control algorithm, as explained in [10] by Yoshizawa et al., is that a reference point is moved along the desired path so that the length between the reference point and the robot is kept at some distance. Control (velocity) commands are generated for the robot to reach that reference position from the current position [10]. This control method, used so far, requires the reference path to be known. But the gradient array of HPF has features which enable the use of this method efficiently without advanced knowledge of the complete path being tracked. The calculated 2-D gradient matrix ($V = [V_x\ V_y]$) contains all the directional information required for the UGV to reach the goal from any point in the workspace. In other words, every point in the region will have an associated guidance vector directing it towards the next *reference position* leading to the goal. This can be taken as the *reference point* required by the quadratic curve fitting controller. Thus, the important region-to-point guidance property of HPF makes it an efficient path planner to be combined with the quadratic curve fitting controller. We do not need to compute the complete path from start to goal before the robot action starts. What we need to compute is the gradient array $V$. The reference position $(x_R, y_R)$ for each current position $(x, y)$ is calculated from the gradient array of the HPF ($\nabla\phi$). Each reference position $(x_R, y_R)$ with respect to $(x_0, y_0)$ is calculated as shown in (7).

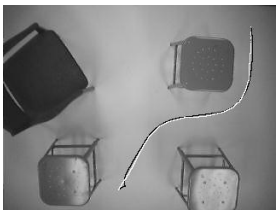
$$\begin{aligned} x_R &= x_0 + V_x(x_0, y_0) \\ y_R &= y_0 + V_y(x_0, y_0) \end{aligned} \qquad (7)$$

During the UGV motion, we can choose the reference position at a discrete look-ahead ($\delta L$) distance from the current position[14]. Eq (7) is calculated in a loop of $\delta L$ iterations to get a reference point at a discrete distance $\delta L$.

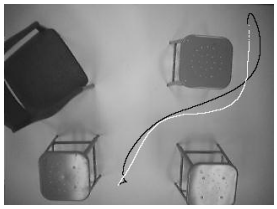
a. *Dynamic look-ahead distance*

The accuracy of the path tracked will be sensitive to the look-ahead distance ($d_0$) selected as the quadratic curve controller will do curve fitting ($y = Ax^2$) between the current and the reference point. The magnitude of the speed ν and the turn-rate ω is proportional to the distance $d_0$ between $(x_0, y_0)$ and $(x_R, y_R)$ [10]. Small $\delta L$ will keep the UGV trajectory close by the desired path. However the time to complete the path will increase because of the small ν and ω (as shown in Fig. 6 and Fig. 7). If $\delta L$ is large, (e.g. $\delta L = 6$) then $d_0$ will be large. 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 will increase and it will increase the probability of hitting nearby obstacles in case of large network delays (Fig. 6(b)). Therefore distance $d_0$ at which the reference point is to be located from the current position and the discrete $\delta L$ can be determined dynamically at each iteration, depending upon the curvature of the path ($An$) at that current position after

fitting the curve $y = An\,x^2$. Eq(8) explains the calculation of $d_0$. $G_D$ is a constant representing the distance at which the gradient value is calculated i.e. distance corresponding to $\delta L$=1. Thus at a high curvature point, $d_0$ will be small, giving rise to a small $\delta L$ (look-ahead distance). The UGV will run slowly being more cautious on the turn. On the other hand if the path is a straight line (low curvature point), then the reference point will be chosen at comparatively a larger distance from current position making the UGV move faster. Thus dynamically choosing the look-ahead distance will incorporate optimality between the path tracking accuracy and the time required to reach the goal. (Fig. 8)



$\delta L$ = 1, Network delay = 0.1mS
T = 27 seconds



$\delta L$ = 8, Network delay = 0.1mS
T = 16 seconds

Fig. 6(a) and (b). Effect of different values of $\delta L$

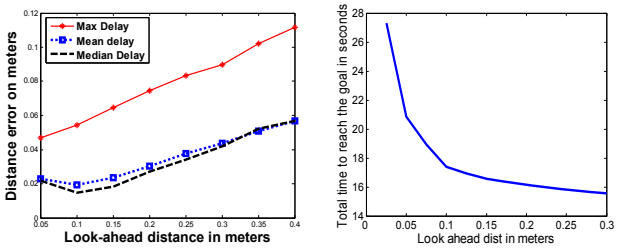
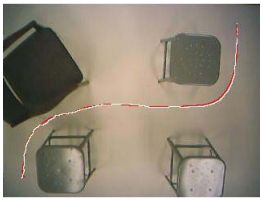



Fig. 7. Distance error increases and total time required to reach the goal decreases with increase in δL.

$$d_0 = \frac{d_{\max}}{1+\beta|An|} \qquad \delta L = \left\lfloor \frac{d_0}{G_D} \right\rfloor \tag{8}$$

$$K_n = sign(e_x)\frac{\alpha}{1+|An|} \qquad v = K_n \qquad \omega = 2\cdot An\cdot K_n$$



Network delay = 0.3 s
T = 24.3 s

Fig. 8: Effect of dynamic look-ahead distance

In the fast marching approach, the complete path coordinates are pre-calculated before the UGV movement starts. E.g. the pre-determined reference path has $N$ discrete points $\{(x_0,y_0),(x_1,y_1),\ldots,(x_N,y_N)\}$ from source to the goal. At each sampling instance $t_i$, the current position of the UGV $(x_0,y_0)$ is determined using template matching. Then each of the $N$ points on the pre-determined path is checked to find out which 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 from the closest point. Thus at each $t_i$, it will take $N$ number of real time checks on the path to find out the closest point and the corresponding reference position. During the tracking, if the feedback loop runs, say, $n$ times, before reaching the goal, then it will cost $(N.n)$ real time computations. When HPF is combined with the quadratic controller, each reference point computation in real time will take $\delta L$ additions. This will take a total of $n.\delta L$ in real time. This surely decreases the real-time computation cost and the time by a factor of $N/\delta L$. The longer the path ($N$) is, the more efficient the HPF planner will be. Thus the path tracking problem is now converted into a goal seeking problem with no pre-determined path to be tracked.

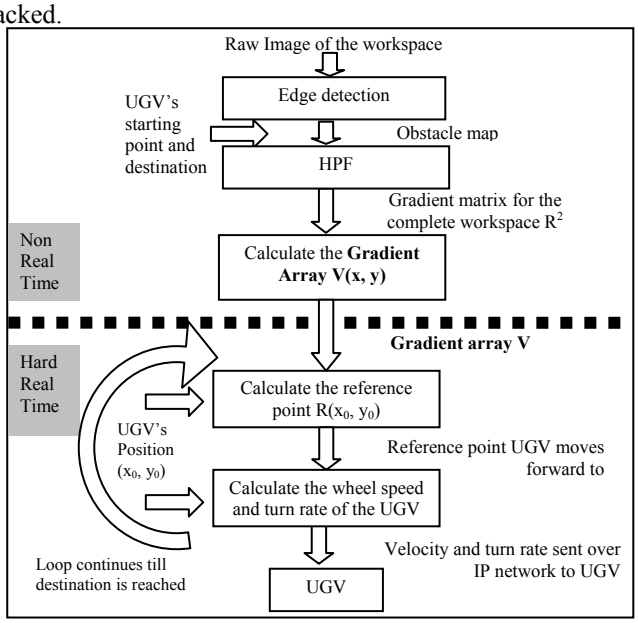



Fig. 9.Complete flow diagram of the proposed system.

For the network delay compensation, we used the gain scheduler middleware technique introduced by Chow and Tipsuwan [14,15]. GSM methodology uses middleware to modify the output of an existing controller based on a gain scheduling algorithm with respect to the current network traffic conditions. The overall GSM operations for networked control and tele-operation are explained in details in [11].

A complete flow diagram for the proposed network based integrated navigation system is shown in Fig. 9.

## III. Results and Discussions

The iSpace workspace described in this paper is a 3m x 4m area. Resolution of the workspace image is 320X240. In case of the HPF planner, though the reference path is not required for the motion controller, we calculated the reference path from source to destination following the negative gradient at each point (δL=1). This represents the path which the UGV will take in case of ideal conditions i.e. (i) no network delay (ii) UGV runs accurately with the speed sent by the controller and (iii) quadratic curve fitted by the controller exactly matches the actual path generated by HPF planner. The capabilities and features of the suggested system are demonstrated using a three-case study. The path taken by the UGV in actuality is compared with the reference path. The closest distance between the UGV's position and the reference path is calculated at each $t_i$. This difference is called the

'distance error'. It shows how close the actual path is to the (ideal) reference 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. Different obstacle and background cases*

Fig. 10 shows that the suggested method works well with different number, shapes and sizes of obstacles in the indoor workspace under consideration. All the turns around the obstacle are at enough safe distance. Without tracking the reference path, the dynamic reference point approach yields the average and the maximum distance error of 2 to 5 cm respectively, which is 1% error with respect to the actual dimension of the space. (Actual workspace-300cm x 400 cm)

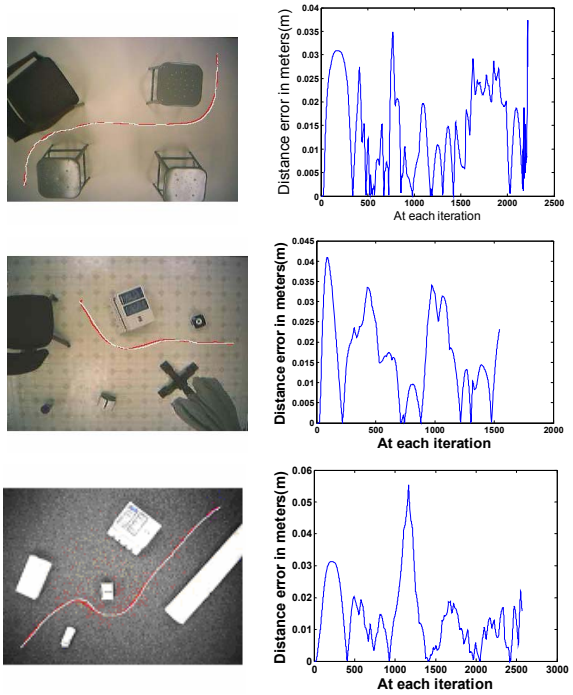



Fig. 10 Actual workspace images and the graph of distance error as a function of time. Red line is actual path and white line is the ideal path.

### *B. Performance with different Network and Processing delay*

Generally the network delay is in the order of few hundreds of millisecond on the Internet. Typical delays observed in US continent are 0.1s to 0.6s. We however tested the functionality for a network delay range of 0.1 s to 1.2 s. Fig. 11 shows that the distance error increases exponentially with increase in network delay. For the average network delay of 0.3 seconds, we have a mean distance error of 1cm and a maximum error less than 5 cm. Even with delays as large as 1.2 sec, the maximum error is 20 cm (4% of the diagonal length of workspace area). Fig. 12 shows that HPF planner provides a smoother obstacle-free path than fast marching method. Graphs in Fig. 12 and Fig. 13 show that the HPF planner acting as a goal-seeker improves the performance of the system in both path and time optimality. Fig. 14 shows the test cases with network delay as high as 1.1 second (more than the network delays experienced on intercontinental communication), the method proves to successfully reach the goal without hitting the obstacle even with such high delay.

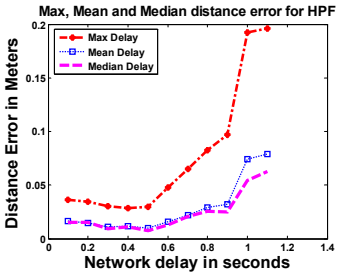



Fig. 11. Mean, Median and Maximum distance error as a function network delay for HPF planner.

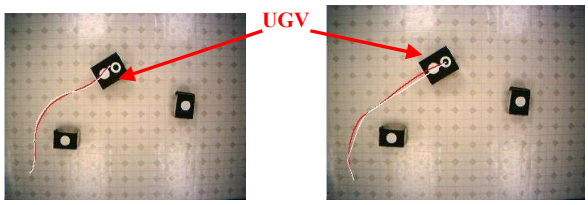



Fig. 1. Left – HPF planner and Right – fast marching planner.

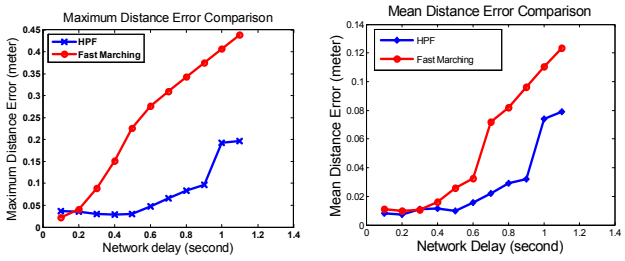



Fig. 12. Maximum and mean distance error vs. network delay comparing fast marching and HPF planner. Dotted red line is for fast marching and continuous blue line is for HPF planner.

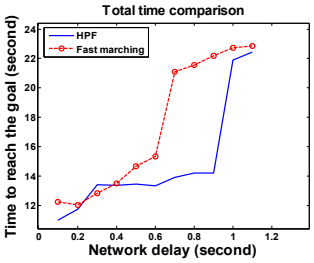
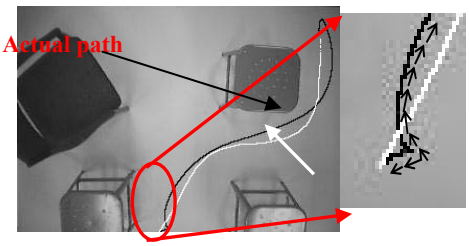



Fig. 13. Total time vs. network delay comparing fast marching and HPF.




The starting point of the UGV is zoomed-in to show the turning of the UGV ($\theta_{init}$ = 225°). Arrows are showing the UGV orientation.

Fig. 14. Navigation results with network delay = 1.1 s and look-ahead distance = 0.4 m. Next zoomed-in image is showing the turning around.

An interesting, intelligent behavior is also 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 towards the target to synthesize a human driver-like maneuver that treats the space as a scarce resource (Fig. 14). The source of this intelligence is the quadratic curve fitting controller. The authors will investigate this hypothesis in the future.

### C. *Special case*

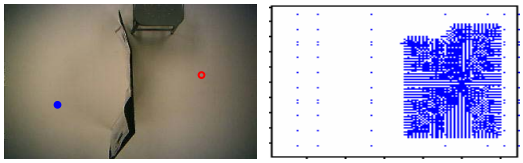


Fig. 2. Image with a barrier separating two regions. Blue dot is the source and red dot is the chosen goal.

HPF has an important property that the potential function in any goal-free closed contour of $\Omega$ will converge to a constant potential [2]. Therefore it displays an intelligent behavior when the goal point chosen is unreachable due to a barrier in the workspace. The complete region around the UGV is pulled up at the equal high potential with $\nabla\phi = 0$ and the UGV does not move from its position as if knowing beforehand that the goal point is unreachable (Fig. 2).

## IV. Conclusion and Future work

The paper successfully shows through experimental results that edge detection, a model based HPF planner and network based quadratic controller go hand in hand to create an efficient and delay-tolerant integrated path planning system giving more generality and flexibility to the robot workspace. This is not the case with template matching. A good edge map helps to build the correct HPF planner representing the obstacles with high potential. The HPF planner is used to generate the reference point dynamically. This not only decreases the computational burden in real time but also achieves optimality between the total time of navigation as well as accuracy of the navigational path, making it more suitable for network-based control. This is evident from the comparison with the fast marching method. The Dirichlet's setting keeps the robot path, as much as possible, away from the obstacles making it efficient even in heavily cluttered environments. The quadratic curve controller displays the intelligent behavior of changing the orientation to face the direction of movement. HPF displays intelligence in special cases such as no movement in case of goal unreachable problems. Thus the new structure suggested satisfies many requirements which are key for a network based integrated navigation system.

Reliable edge detection is the backbone of the vision module for the obstacle boundary detection as a part of the NBINS suggested in this paper. The harmonic potential planner uses a raw edge map to generate the guidance field without any need to interpret the map's contents. There are several difficulties that could arise when the edges of a non-synthetic image are to be detected. Noise, shadows, variable texture or contrasts in the image could all cause false edge components. Since no side information exists in an edge map, to the robot, a false edge is as real as a true edge. This could make the robot behave erratically. There is also the danger of losing edge contours corresponding to physical environment contents. In this case collision could occur. The workspace is illuminated from the top, shadows will lie close to the obstacles and we may safely leave them. However, the story will be different if a side light source is present. Therefore currently the system is suitable for indoor environments with control over the light source and illuminations. This system surely shows potential for indoor applications like nursing homes, manufacturing plants etc. The authors will continue to investigate and enhance the prepared techniques so that it can be used effectively even in an unfavorable environment.

## Reference


[1] Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. in Robotics and Automation. Proceedings. 1985 IEEE International Conference on. 1985.

[2] Connolly, C.I., J.B. Burns, and R. Weiss. Path planning using Laplace's equation. in Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on. 1990.

[3] Connolly, C.I. Applications of harmonic functions to robotics. in Intelligent Control, 1992., Proceedings of the 1992 IEEE International Symposium on. 1992.

[4] Connolly, C.I. and R.A. Grupen. Harmonic control [robot applications]. in Intelligent Control, 1992., Proceedings of the 1992 IEEE International Symposium on. 1992.

[5] Kim, J.-O. and P.K. Khosla, Real-time obstacle avoidance using harmonic potential functions. Robotics and Automation, IEEE Transactions on, 1992. 8(3): p. 338-349.

[6] Tanner, H.G. and K.J. Kyriakopoulos. Nonholonomic motion planning for mobile manipulators. in Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on. 2000.

[7] Masoud, S.A. and A.A. Masoud, Motion planning in the presence of directional and regional avoidance constraints using nonlinear, anisotropic, harmonic potential fields: a physical metaphor. Systems, Man and Cybernetics, Part A, IEEE Transactions on, 2002. 32(6): p. 705-723.

[8] Alvarez, D., J.C. Alvarez, and R.C. Gonzalez. Online motion planning using Laplace potential fields. in Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on. 2003.

[9] Wesley Snyder, Hairong Qi, Machine Vision, Cambridge University Press, 2004

[10] K. Yoshizawa, H. Hashimoto. M. Wada, and S. M. Mori, Path tracking control of mobile robots using a quadratic curve, presented at IEEE Intelligent Vehicles Symposium'96, Tokyo, Japan, 1996.

[11] Tipsuwan, Y., Chow M.-Y., Gain scheduler middleware: a methodology to enable existing controllers for networked control and tele-operation - part I: networked control, IEEE Transactions on Industrial Electronics Dec. 2004 Volume 51, Issue 6, Page(s):1218 – 1227

[12] Wai-Lun Danny Leung, Rangsarit Vanijjirattikhan, Zheng Li, Le Xu, Tyler Richards, Bulent Ayhan, Mo-Yuen, Chow, Intelligent Space with Time Sensitive Applications, Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics

[13] Sethian, J. A., A fast marching level set method for monotonically advancing fronts. Proceedings of the National Academy of Science of the United States of America, February 1996, PNAS 93, 1591 - 1595.

[14] R A Gupta, A. Masoud and M. Y. Chow, A Delay-tolerant, Potential field-based, Network Implementation of an Integrated Navigation System in Proceedings of the IEEE/RSJ International conference on Intelligent Robots and Systems, 2006.