ABSTRACT

OJHA, UNNATI. Delay Tolerant Behavior Control based Adaptive Bandwidth Allocation for Unmanned Ground Vehicles in a Network Control System. (Under the direction of Dr. Mo-Yuen Chow)

The rapid growth of wired as well as wireless communication technology has led to the expansion of Network Control System (NCS) into Large Scale Distributed Network Control System. As the size of NCSs increase, important resources like bandwidth become scarce which degrade the performance of the NCS. It is necessary to manage the bandwidth in such a way that causes the least amount of degradation in the NCS performance. Many adaptive sampling methods have been suggested to meet the bandwidth constraint; however, efficiency of these methods decreases quickly as the amount of network delay increases. Delay in the network is unavoidable and stochastic thus it is important that a bandwidth management technique be robust to delays in order to maintain the performance of the NCS. This study proposes a Delay Tolerant Behavior Control (DTBC) based bandwidth allocation that is robust to network delays. This new method for bandwidth allocation in a system of Unmanned Ground Vehicles (UGVs) is inspired by the human driving behavior. In DTBC, bandwidth is allocated based on the UGV trajectory curvature, amount of network delay and the velocity of the UGV. A study is first presented to analyze the relationship between curvature and the UGV performance. The Behavior Control (BC) based bandwidth allocation algorithm, a predecessor of DTBC is then formulated. The BC algorithm is then expanded and improvised to formulate the DTBC algorithm. Both simulation and implementation results are presented at the end which confirms that DTBCs performance is superior to the existing algorithms especially in the presence of network delay.

Delay Tolerant Behavior Control based Adaptive Bandwidth Allocation for Unmanned Ground Vehicles in a Network Control System

by
Unnati Ojha

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Electrical Engineering

Raleigh, North Carolina

July 2009

APPROVED BY:

_____	_____
Dr. Huaiyu Dai	Dr. James J. Brickley

_____
Dr. Mo-Yuen Chow
Chair of Advisory Committee

BIOGRAPHY

Unnati Ojha was born in Nepal on the 25th of June, 1982. He received his Bachelor of Engineering degree in Electrical Engineering with a minor in Computer Engineering in 2004 from College of Engineering and Agro-Industrial Technology, University of the Philippines at Los Baños, Philippines. After his graduation, he went back to his country and worked as a Lecturer in Kathmandu Engineering College (KEC), Kathmandu, Nepal. He also supervised the Robotics and Automation Club at KEC, which was able to win the national level Robotics Competition for two consecutive years under his supervision. In fall of 2007, he joined North Carolina State University as a graduate student. He has been working under the guidance of Dr. Mo-Yuen Chow at the Advanced Diagnosis Automation and Control (ADAC) lab since spring 2008.

His primary research interest lies in network based control and automation of time-sensitive applications. He hopes to see commercial, fully-automated cars during his lifetime. Currently he is working at ADAC lab on a project which involves real-time optimization and resource allocation in large scale time sensitive network control system and aims at reducing traffic accidents and helping impaired drivers. With the defense of this thesis, he is receiving the degree of Master of Science in Electrical Engineering from NCSU.

ACKNOWLEDGMENTS

I would first like to extend my heartfelt gratitude to Dr Mo-Yuen Chow for his invaluable guidance and persistent support during the course of this research. I am thankful for him for providing me with a platform to work on my interests. His suggestions and directions have greatly helped me build up my research as well as professional capabilities. I have learnt and realized the importance of planning and organization through him and I am very grateful for that.

In am thankful to Dr James J. Brickley and Dr Huaiyu Dai for kindly agreeing to be on my thesis committee.

I would like to thank my colleague Bryan Klingenberg with whom I have been collaborating to develop the latest version of iSpace platform and the iSpace simulator. Our discussions have contributed a lot in the technical aspects of the implementation of this study.

I also acknowledge the help of my lab members: Xiao Feng Pang, Preetika Kulshrestha, Yixin Cai, Rachana Gupta, Cameron Lynch, Ben Stiles and Hanlei Zheng who were always ready to help me out whenever I faced a problem.

None of this would have been possible without my family. They have been a source of tremendous strength and inspiration. I would also like to thank Ms Biva Shrestha for always being there to motivate me when I lost my focus and lighten me up when I felt frustrated. Last but not the least I would like to thank all my friends who have been there when I needed them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1- Introduction

1.1 Background

Last few decades have marked a tremendous increase in the deployment of wired and wireless networked systems in business, industry as well as academia which have accelerated the research and development in the area of Distributed NCS. Network Control Systems (NCS) refer to closed loop control system where feedback and/or control signals are transmitted over a communication channel that may be used by other local or remote systems. A NCS can be classified as a hierarchical or a direct structure. In a hierarchical structure, the controlled subsystems (sensors and actuators) are directly connected to the supervisory controller whereas in direct structure each subsystem has its own local controller that in turn communicates with the supervisory controller [1].

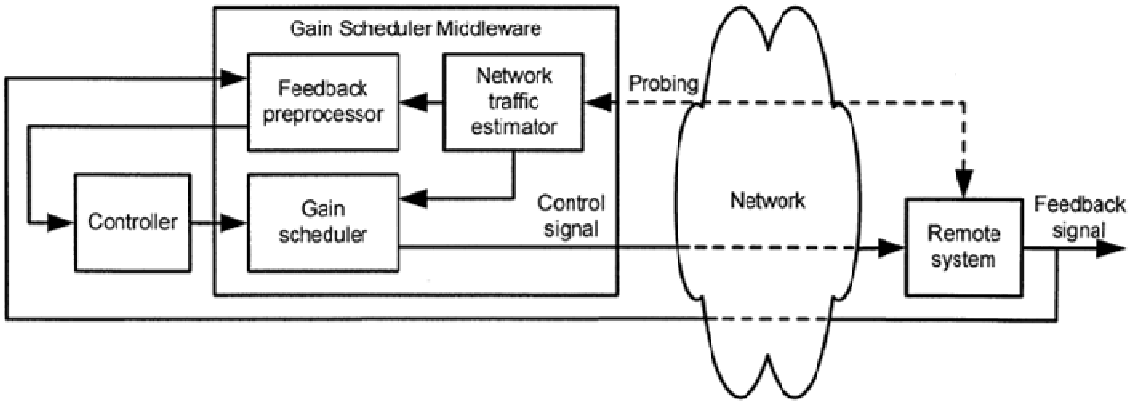


Figure 1: A typical NCS with a single remote plant

Some application areas for NCS include intelligent transportation systems, space explorations, factory automation, cyber-physical systems, and semi-automated nursing

homes etc. We can broadly categorize NCS applications into two categories as (1) time-sensitive applications or time-critical control such as military, space and navigation operations; (2) time-insensitive or non-real-time control such as data storage, sensor data collection, e-mail, etc[2]. Time sensitive applications can be further divided into hard real-time sensitive applications where the reward is maximum if the intended deadline is met and minimum otherwise and soft real-time sensitive applications where the reward decreases slowly if we miss the deadline.

Figure 2 shows the difference between hard and soft real-time applications.

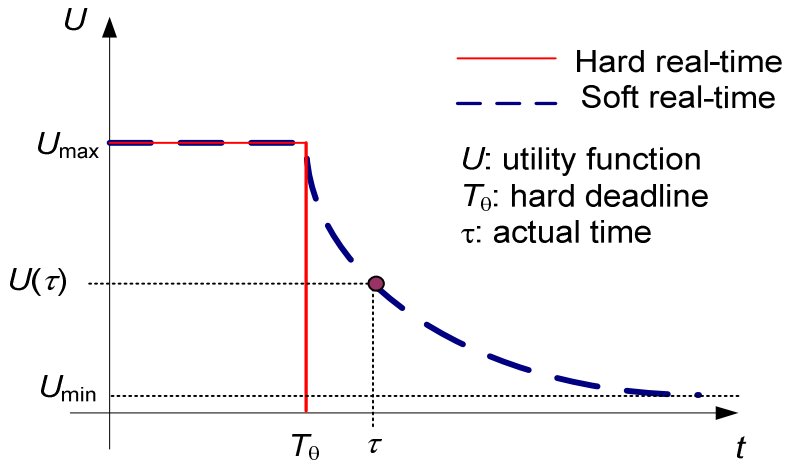


Figure 2: Types of time sensitive applications

1.2 Intelligent Space (iSpace)

Intelligent space (iSpace) is a relatively new concept to effectively use various engineering disciplines such as automation and control, hardware and software design, image processing, distributed sensors, actuators, robots, computing processors and information technology over communication networks over a space of interest to make intelligent

operation decisions. It can also be considered as a large-scale mechatronic system over networks. This space can be as small as a room or a corridor or can be as big as an office, city or even a planet[3]. Advanced Diagnosis Automation and Control lab at North Carolina State University in Raleigh has developed a multi-sensor network controlled integrated navigation system for multi-robots demonstrating the concept of iSpace[4].

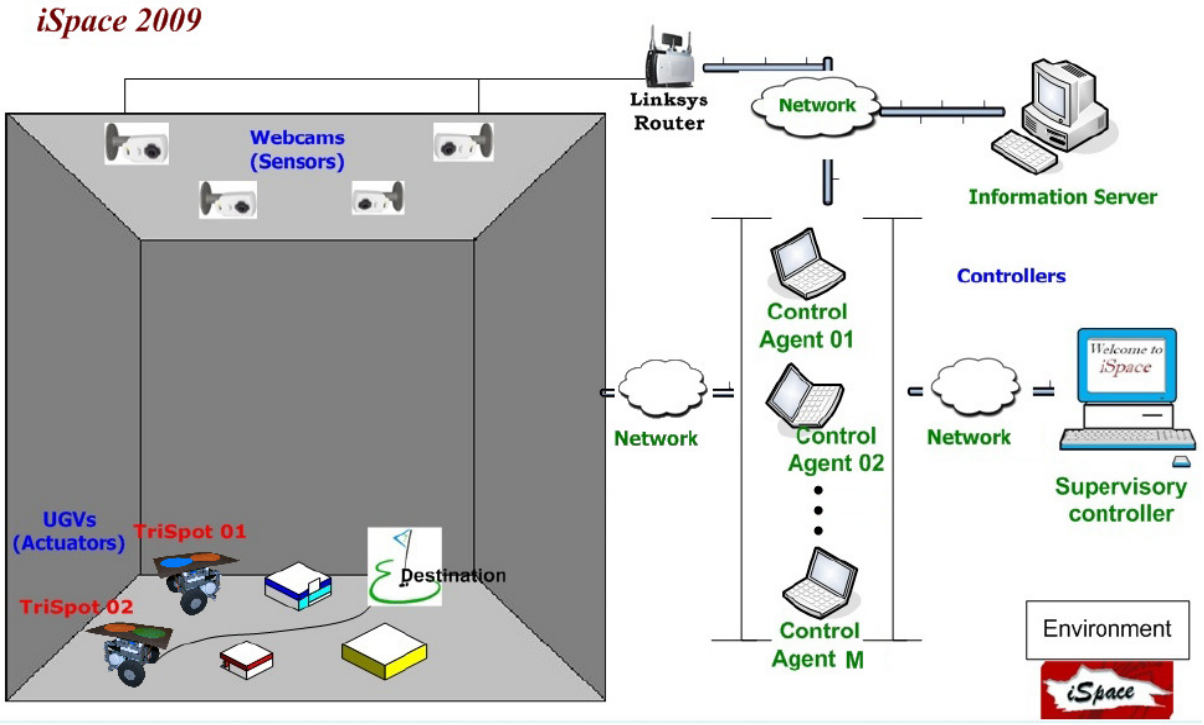


Figure 3: iSpace at ADAC

This iSpace a useful platform for studying distributed network control systems, specifically the networked control of Unmanned Guided Vehicles (UGVs) operating within iSpace. The latest version of iSpace updated in 2009 has been built on previous versions reusing some code and modules and adding new modules and platforms. In this new version of iSpace a new supervisory controller, distributed controllers, and new UGVs have been

developed. This iSpace can be used to study the effects of various network constraints and ways to optimize the UGV performance under those constraints.

1.3 Issues in Network Control Systems

As the concept of NCS started to grow because of its potential in various applications, it also provided many challenges for researchers to achieve reliable and efficient control. Some of the current research topics under NCS include the different control strategies and kinematics of the actuators/vehicles suitable for NCS, the study of the network structure required to provide a reliable, secured communication channel with appropriate bandwidth management, compensation of time-delay issues in the network, the development of data communication protocols for control systems, and the efficient collection of real-time information over a network using distributed sensors [2]. In real-time NCSs, the stochastic nature of network time-delay is one of the most important problems.

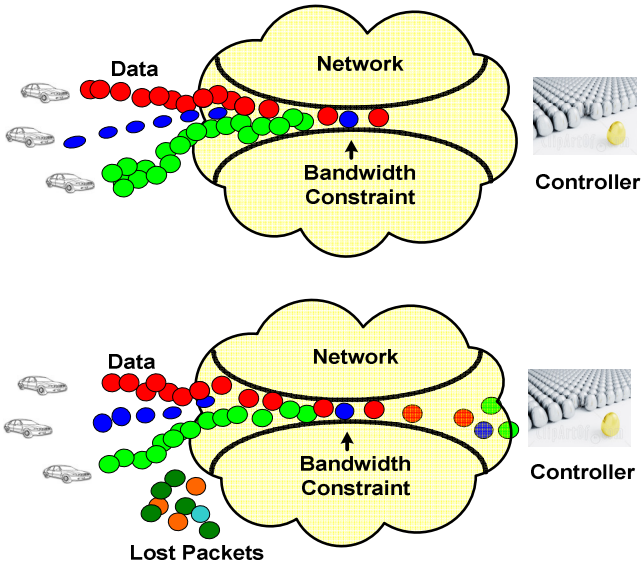


Figure 4: Bandwidth constraint and its effect in the network

As the size of NCS becomes larger and multiple plants are controlled through the network, in addition to the time-delay issue, the bandwidth constraint also becomes an important factor for the performance degradation. Xia [5] says that network delay is not the only factor that degrades the performance of the NCS. Some other constraints like network bandwidth and CPU processing time may cause the quality of control to degrade to an extent that it may cause instability. Xia, in the same work has also suggested that bandwidth allocation can be an effective method to improve performance of control loops under such conditions. Since the network closes the feedback loop in a NCS, it becomes important for us to sample the feedback from the remote plant only when necessary because of this bandwidth constraint. Figure 4 shows that because of the limited amount of bandwidth not all data can be sent from the plants to the controller. If bandwidth is not managed properly, we will lose important data because of the congestion.

1.4 Behavior Control Based Approach for Bandwidth Allocation

This study focuses on developing a novel adaptive bandwidth allocation method, called the Behaviour Control (BC) Approach, in a multiple UGV system controlled through the network. The methodology developed here is inspired by the human driving behaviour. An analogy between the amount of attention the human drivers give and the amount of bandwidth required is taken as the foundation for this work. Bandwidth is considered as a limited resource and is allocated to an UGV in accordance to the curvature of its trajectory and its velocity relative to the other UGVs. Chapter 2 discusses the mathematical foundation that establishes the relationship between curvature of the road and the performance of a

Kalman Filter. Chapter 3 presents the first attempt at relating bandwidth with the curvature and velocity of the vehicle. Chapter 4 further improves the algorithm developed in Chapter 3 to accommodate network delay thus formulating a Delay Tolerant BC approach. Chapter 5 discusses the results of the implementation of the developed algorithm in iSpace at ADAC. Conclusion of this study is presented at the end.

References

[1] M.-Y. Chow and Y. Tipsuwan, "Network-based control systems: a tutorial," in *Industrial Electronics Society, 2001. IECON '01. The 27th Annual Conference of the IEEE*, 2001, pp. 1593-1602 vol.3.

[2] R. A. Gupta and M. Y. Chow, "An Overview of Networked Control Systems," in *Networked Control Systems: Theory and Applications*, F.-Y. Wang and D. Liu, Eds.: Springer-Verlag, 2007.

[3] R. Vanijjirattikhan, M.-Y. Chow, P. Szemes, and H. Hashimoto, "Mobile Agent Gain Scheduler Control in Inter-Continental Intelligent Space," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, 2005, pp. 1115-1120.

[4] W. L. D. Leung, R. Vanijjirattikhan, L. Zheng, X. Le, T. Richards, B. Ayhan, and M.-Y. Chow, "Intelligent space with time sensitive applications," in *Advanced Intelligent Mechatronics. Proceedings, 2005 IEEE/ASME International Conference on*, 2005, pp. 1413-1418.

[5] F. Xia, W. Zhi, and S. Youxian, "Simulation based performance analysis of networked control systems with resource constraints," in *Industrial Electronics Society, 2004. IECON 2004. 30th Annual Conference of IEEE*, 2004, pp. 2946-2951 Vol. 3.

Chapter 2- Analysis on the Kalman Filter Performance in GPS/INS Integration at Different Noise Levels, Sampling Periods and Curvatures

Unnati Ojha, Mo-Yuen Chow
North Carolina State University
uojha,chow@ncsu.edu

Timothy Chang, Janice Daniel
New Jersey Institute of Technology
chang,daniel@njit.edu

This chapter has been submitted for publication to the 35th Annual Conference of the IEEE Industrial Electronics Society (IECON '09), Porto, Portugal, November, 2009.

2.1 Abstract

Kalman filters (KF) have been extensively used in the integration of Global Positioning System (GPS) and Inertial Navigation System (INS) data. Often, the GPS data is used as a benchmark to update the INS data. In this study, an analysis of integration of GPS data with INS data using an Extended Kalman filter is performed in terms of the filter's performance with respect to the amount of noise in the GPS data and the sampling time of the vehicle position. The study further analyzes and compares the pattern of error at varying sampling periods in vehicle trajectories with high curvature path segments and low curvature path segments. Simulation results are presented at the end. The results show that the performance of the KF depends linearly on the amount of noise and sampling times. The relationship between the curvature of the road and the performance of the KF was found to be quadratic.

2.2 Introduction

In navigation systems, Global Positioning System (GPS) has been widely used for vehicle localization because of its low-cost and world-wide coverage. The accuracy of the measurements in these GPS units varies according to the cost and the number of satellites the GPS unit can connect to at any instant. The accuracy of GPS can be improved using different algorithms like Wide Area Augmentation System (WAAS), Differential GPS, etc. The accuracy can range from 1m in differential GPS up to ~10m in WAAS. However, depending on the number of satellites that the unit is connected to, the accuracy can easily worsen to as much as 30m (100 ft).

The Inertial Navigation System (INS) uses dead reckoning to calculate the position of the vehicle and can provide position and angle updates at a quicker rate than GPS. However, in INS, small measurement errors in acceleration and angular velocity are integrated into progressively larger errors in velocity, which eventually causes greater errors in position. INS systems can be used with some other system like the GPS to provide a better accuracy measurement. Kalman Filter is commonly used in order to fuse the information from GPS and INS.

In applications related to advanced congestion management and planning, data about the position and the orientation of the vehicle should be consistent and precise. Bhavsar, Chang, Daniel and Chow [1] have used pseudo-GPS with image-captured data in intelligent traffic management systems. In designing a system for these applications, it is important to find out the performance of the system at different operating conditions. In studies like [2, 3] by Zheng and Chow and [4] by Ojha, Klingenberg and Chow, it is important to understand the relationship between sampling rate, road curvature and the system performance in order to build an efficient bandwidth allocation scheme for intelligent transportation management. Several studies have been done on improving the performance of GPS signals and integrating the GPS data with the INS [5-8]. Wei and Schwarz use two Kalman filters separately to filter the GPS signals and then use the filtered GPS signal as updates to the estimates obtained from the inertial data in [5]. Similarly, Chaffee, Abel, and McQuiston [6] survey the problems encountered in the statistics of point estimation of position and bias from GPS solutions. Both Gao et. al. [7] and Hide [8] use adaptive Kalman filtering in order to compensate for the unpredictable amount of measurement noise in GPS.

While these studies focus on introducing different types of Kalman filters as a tool for better estimation, some other studies have looked into the issues that come with using these filters. Schneider and Maida [9] have done a study on integrating GPS signals with Doppler radar navigation system. In [10], Smith et al. have analyzed and developed the general sufficiency conditions for the stability of a discrete-time cascaded filter architecture. Seong [11] has analyzed the observability of the INS/GPS navigation system for a land vehicle on measurements and different dimension filters with respect to the measurements. In [12], Geng and Wang focus on developing a new approach to adaptive estimation of multiple fading factors in the Kalman filter for navigation. The proposed approach is based on the assumption that, under optimal estimation conditions, the residuals of the Kalman filter are Gaussian white noises with a zero mean. In [13], Kibangou and Monin have derived a nonlinear and non-Gaussian model as the process model using GPS pseudo-range and odometer measurements.

In this paper, we use an Extended Kalman Filter (EKF) in order to fuse the data from GPS and INS. The uniqueness of the study lies in the analysis of using EKF in this data fusion model at different noise levels, sampling rates and road curvatures. The study provides the relationship of the performance of the EKF with respect to varying road curvatures sampling rates and noise levels.

This paper is organized in the following manner. Section II briefly discusses the system and derives the necessary parameters for the developed data fusion model. The approach taken for the analysis of the developed model in terms of noise levels, sampling

times and curvatures is explained in section III. Section IV presents the simulation results and the paper is concluded in section V.

2.3 System Description

A two-state vehicle dynamics model along with the Extended Kalman Filter was used to fuse the data from GPS and INS. The GPS data consisted of the latitude (*Lat*) and the longitude (*Lon*) of the vehicle measured in *degrees, minutes* and *seconds* following the WGS84 (World Geodetic System, 1984) standard. The GPS also gives the altitude (*h*) of the vehicle which is measured in *meters*. The inertial measurements included the velocity (*v*) and orientation (θ) of the vehicle measured in *m/s* and *degrees* respectively.

In order to fuse the data properly, the data from GPS and INS need to be consistent with each other in terms of the measurement units used. The GPS coordinates in the WGS84 form should be converted to ECEF (Earth Centred Earth Fixed) Coordinates using:

$$\begin{aligned} x_{ECEF,k} &= \left(\frac{a}{\chi_k} + h_k\right)\cos\left(Lat_k\right)\cos\left(Lon_k\right), \\ y_{ECEF,k} &= \left(\frac{a}{\chi_k} + h_k\right)\cos\left(Lat_k\right)\sin\left(Lon_k\right), \\ z_{ECEF,k} &= \left(\frac{a\left(1-e^2\right)}{\chi_k} + h_k\right)\cos\left(Lat_k\right)\sin\left(Lon_k\right), \end{aligned} \qquad (2.1)$$

where a is earth's semi-major axis in meters (a=6378137 m). e^2 is the square of the first numerical eccentricity of the ellipsoid (e^2 = 6.69437999014x10^{-3}). h_k is the altitude of the vehicle. χ_k is calculated using:

$$\chi_k = \sqrt{1-\left(2\times rf - rf^2\right)\times \sin^2\left(Lat_k\right)}, \qquad (2.2)$$

where *rf* is the reciprocal flattening (*rf* = 0.0033528). The ECEF measurements obtained after using (2.1) are then changed to the Local Tangential Plane Coordinates by:

$$\begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} = \Phi \begin{bmatrix} x_{ECEF,k} - xref_{ECEF} \\ y_{ECEF,k} - yref_{ECEF} \\ z_{ECEF,k} - zref_{ECEF} \end{bmatrix} \qquad (2.3)$$

$$\Phi = \begin{bmatrix} -\sin Lon_k & \cos Lon_k & 0 \\ -\sin Lat_k \cos Lon_k & -\sin Lat_k \sin Lon_k & \cos Lat_k \\ \cos Lat_k \cos Lon_k & \cos Lat_k \sin Lon_k & \sin Lat_k \end{bmatrix} \qquad (2.4)$$

where ($xref_{ECEF}$, $yref_{ECEF}$, $zref_{ECEF}$) is the ECEF coordinate of the reference point which in our study is located at (0,0,0) in WGS84 coordinate system. (x_k, y_k, z_k) in (2.3) are in meters and are consistent with the INS measurements. The data from GPS and INS is then fused by using the vehicle dynamics given by:

$$\begin{aligned} x_{k+1} &= x_k + v_k \cos(\theta_k)\Delta T, \\ y_{k+1} &= y_k + v_k \sin(\theta_k)\Delta T, \end{aligned} \qquad (2.5)$$

where (x_k,y_k) calculated using (2.3), represent the position of the vehicle at time k, and v_k and θ_k represent the velocity and orientation measurements at time k respectively. ΔT is the sampling interval. A number of techniques are available to convert from ENU to WGS84. The one that was used in this study involves a series of steps which is described by Zhu in [14].

As we can see in (2.5), the system model is non-linear and requires the use of EKF to linearize the model at each operating point before we can use the linear Kalman filter equations to estimate the optimal gain. The system in (2.5) can be expressed as:

$$\begin{aligned} \mathbf{x}_{k+1} &= f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k), \\ \mathbf{z}_k &= \mathbf{H}\mathbf{x}_k + \boldsymbol{\upsilon}_k, \end{aligned} \quad (2.6)$$

where

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}, \ \mathbf{u} = \begin{bmatrix} v \\ \theta \end{bmatrix}.$$

$\boldsymbol{w}$ and $\boldsymbol{\upsilon}$ are the process and measurement noise respectively. Using EKF, the state variables can be estimated using the time update equations given by:

$$\hat{\mathbf{x}}_{k+1}^{-} = f(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0), \quad (2.7)$$

$$\mathbf{P}_{k+1}^{-} = \mathbf{A}\mathbf{P}_k\mathbf{A}^T + \mathbf{Q}_k, \quad (2.8)$$

and the measurement update equations are:

$$\mathbf{K}_{k+1} = \mathbf{P}_{k+1}^{-}\mathbf{H}^T(\mathbf{H}\mathbf{P}_{k+1}^{-}\mathbf{H}^T + \mathbf{R})^{-1}, \quad (2.9)$$

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_{k+1}^{-} + \mathbf{K}(\mathbf{z}_{k+1} - \mathbf{H}\hat{\mathbf{x}}_{k+1}^{-}), \quad (2.10)$$

$$\mathbf{P}_{k+1} = (\mathbf{I} - \mathbf{K}_{k+1}\mathbf{H})\mathbf{P}_{k+1}^{-}, \quad (2.11)$$

where

$$\mathbf{A} = \frac{\partial f}{\partial \mathbf{x}}\left(\hat{\mathbf{x}}_\mathbf{k}, \mathbf{u}_\mathbf{k}, 0\right) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \quad (2.12)$$

$$\mathbf{H} = \frac{\partial \mathbf{H}}{\partial \mathbf{x}}\left(\hat{\mathbf{x}}_k, 0\right) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \quad (2.13)$$

P needs to be initialized based on the initial estimate of **x**. Since we are using the GPS data to initialize **x**, **P** can be initialized according to the amount of noise in the GPS measurement. The process noise **Q** and measurement noise **R** also depend on the amount of noise in the GPS and INS data.

2.4 System Analysis at Various Levels of Noise, Sampling Rate and Curvature

i) Analysis using Varying Noise Levels

Nomenclature:
Lat_k = Actual latitude at time k
Lon_k = Actual longitude at time k
ΔLat_k = Amount of noise added to Lat_k
ΔLon_k = Amount of noise added to Lon_k
$\tilde{L}at_k$ = Measured latitude at time k
$\tilde{L}on_k$ = Measured longitude at time k
$\sigma_{Lat,i}$ = Standard deviation of noise ΔLat_k at i^{th} iteration
$\sigma_{Lon,j}$ = Standard deviation of noise ΔLon_k at j^{th} iteration

The EKF performance was evaluated under various noise levels. A clean set of data was produced using the information from Google Earth. In the clean data set, Gaussian white noise was added to both latitude and longitude values with varying standard deviations ranging from 2m to 8m at 0.5m interval.

The amount of noise to be added was calculated using(2.14).

$$\Delta Lat_i = \frac{N(0,\sigma_{Lat,i})}{111034.61}, \quad \Delta Lon_j = \frac{N(0,\sigma_{Lon,j})}{85393.83}, \qquad (2.14)$$

where ΔLat_i and ΔLon_i are 1xn arrays of noise to be added. The values in the denominator are the factors that convert the distance in meters to a change in the degrees of latitude or longitude. $N(\mu,\sigma)$ represents a Gaussian distribution with mean μ and standard deviation σ. The standard deviation values were calculated as:

$$\sigma_{Lat,i} = 0.5\times i, i \in \left[0,..,20\right],$$
$$\sigma_{Lon,j} = 0.5\times j, j \in \left[0,..,20\right].$$

The values calculated in (2.14) were then added to the clean data set to get the noisy longitude and latitude values as:

$$\begin{aligned} \tilde{Lat}_k &= Lat_k + \Delta Lat_k, \\ \tilde{Lon}_k &= Lon_k + \Delta Lon_k. \end{aligned} \tag{2.15}$$

During the evaluation, the measurement noise (R) and process noise (Q) used are:

$$\mathbf{R}_{i,j} = \begin{pmatrix} \sigma^2_{Lat,i} & 0 \\ 0 & \sigma^2_{Lon,j} \end{pmatrix}, \mathbf{Q} = \begin{pmatrix} 0.25 & 0 \\ 0 & 0.25 \end{pmatrix}.$$

The value of Q determines the amount of weight given to the measurement the system. In our system, a high value for Q (~1.5) yielded in estimates that were very close to the noisy signal, thus constraining the improvement in accuracy. Very low values for Q (< 0.1) smoothened the estimates so much that even some significant corners and curves in the actual path were missed, thus again constraining the improvement in accuracy. Choosing Q within these two bounds did not considerably affect the performance of the EKF. This constant value was chosen because it yielded the least error at the scenario when the noise had a standard deviation of (5, 5) and data were sampled at 1Hz.The noisy data from (2.15) was then used as the GPS input and was fused with the INS data using the EKF model.

ii) Analysis using Different Sampling Rates

The clean set of data were generated from Google Earth, at 10m intervals, which corresponds to a sampling rate of 1 Hz for a vehicle moving at a speed of 10m/s. In our evaluation, the vehicle was moving at 10m/s so the minimum sampling

time possible with the given data was 1s. In order to obtain lower sampling rates, the data points were taken at every N^{th} interval where N is the sampling time in seconds. The data at lower sampling rates were obtained using:

$$\mathbf{xd}_n = \mathbf{x}_{N\times n}, n \in \left[1, 2, ..., \left\lfloor \frac{m}{N} \right\rfloor\right], \quad (2.16)$$

where **xd** represents the down-sampled data set, **x** is the actual data set, N is the sampling interval and m is the total number of data points.

However, in this case, the model described in section II was modified into an Adaptive Kalman Filter by placing a scaling factor in the calculation of the a priori error covariance shown in (2.8). Equation (2.17) reflects the modification in the calculation.

$$\mathbf{P}_{k+1}^{-} = S(\mathbf{A}\mathbf{P}_k\mathbf{A}^T + \mathbf{Q}_k), \quad (2.17)$$

where S is the scaling factor. Different values of the scaling factor were used and the best candidate for a particular sampling period was calculated. This process was repeated at different levels of noise to verify the consistency of the performance. Equation (2.18) shows the calculation of S for a given sampling time where S is the scaling factor that minimizes the tracking error.

$$\min_{S\in(0,1)} \bar{e}(\hat{\mathbf{x}}, \mathbf{x}), \quad (2.18)$$

where $\hat{x}$ is calculated using (2.10) and **K** is calculated using (2.9) and (2.17). The error $\bar{e}$ is the mean tracking error given by(2.21). Optimal S was obtained numerically.

iii) Analysis using Different Curvature Roads

In order to understand the relationship between the road curvature and the vehicle position estimation, the study was conducted in segments of paths with radii ranging from 100m to 1000m at 100m intervals. Figure 5 shows the test paths and the radius (R) of the paths.

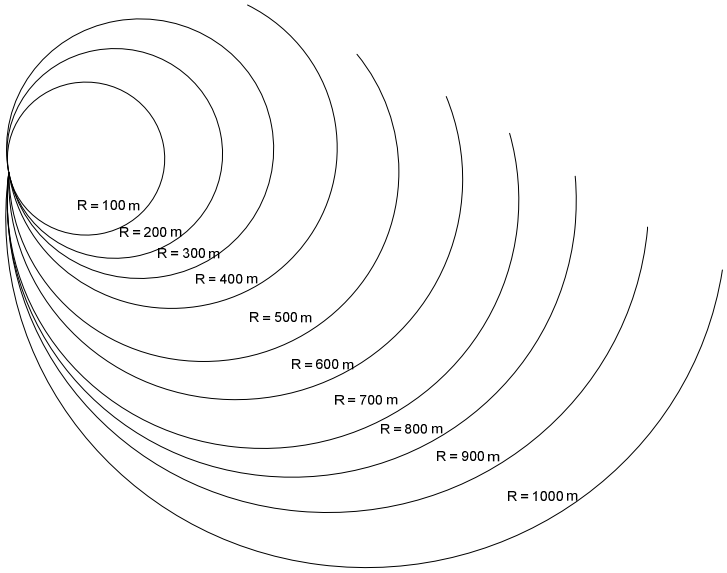


Figure 5. The test paths with segments of different curvature

The curvature is calculated as:

$$\kappa = \frac{1}{R}. \qquad (2.19)$$

iv) Performance Evaluation

The performance of the data fusion model developed was evaluated in terms of the mean and the maximum trajectory tracking error. The trajectory estimation error e_k is the Euclidean distance between the estimated position and the actual position of the vehicle and is given by:

$$e_k = \sqrt{\left\{\left(\hat{L}at_k - Lat_k\right)^2 + \left(\hat{L}on_k - Lon_k\right)^2\right\}}. \qquad (2.20)$$

The mean trajectory tracking error $\bar{e}$ for each of the datasets was calculated as:

$$\bar{e} = \frac{1}{n}\sum_{k=1}^{n} e_k. \qquad (2.21)$$

The mean error was used because it represented the average distance that the estimated position deviated from the actual vehicle trajectory during a particular run. This gives us the overall accuracy of the model.

2.5 Results and Discussion

Figure 6 shows the mean path tracking error before data fusion. Figure 7 shows the mean path tracking error for a vehicle at a speed of 10m/s after data fusion. The sampling time was 1 Hz. The extreme error values range from 0m to 3.67m. The accuracy was increased by up to 63% as compared to the unfiltered data. The mean error shows a growing trend as the noise level in the measurements increases from a standard deviation of (0, 0) to (10, 10).

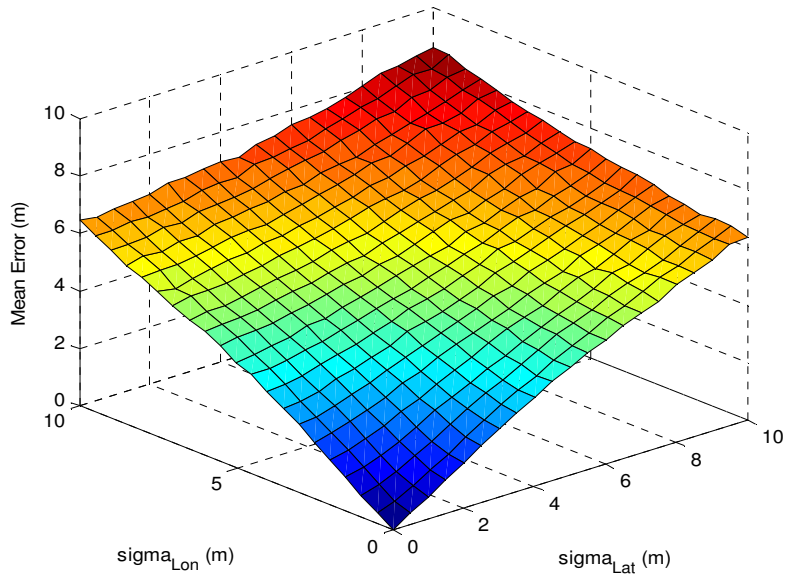


Figure 6. Mean path tracking error for vehicle (v=10m/s, Ts = 1s) at different levels of noise BEFORE combining GPS and INS.

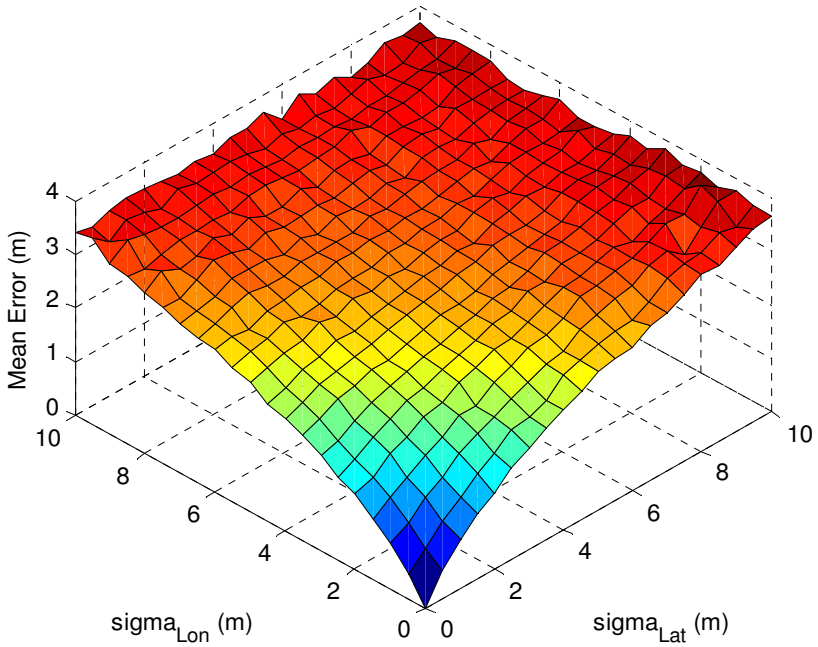


Figure 7. Mean path tracking error for vehicle (v=10m/s, Ts = 1s) at different levels of noise (after data fusion).

Figure 8 shows the improvement in the accuracy of the estimation after using the data fusion model. The increase in the performance accuracy was calculated by:

$$\% \ improvement = \frac{\tilde{e} - \bar{e}}{\tilde{e}} \times 100, \qquad (2.22)$$

where

$$\tilde{e} = \sqrt{\left(\tilde{L}at_k - Lat_k\right)^2 + \left(\tilde{L}on_k - Lon_k\right)^2}. \qquad (2.23)$$

Though the amount of error increases as the noise level increases the improvement in the accuracy given by (2.22) as compared to the noisy data is higher as the noise level increases. The accuracy is improved by up to 63% as compared to the noisy data. There is a sharp increase in the performance improvement at the lower noise levels ((σ_{Lat}, σ_{Lon}) < (2, 2)). At higher noise levels, the performance improvement is uniform .

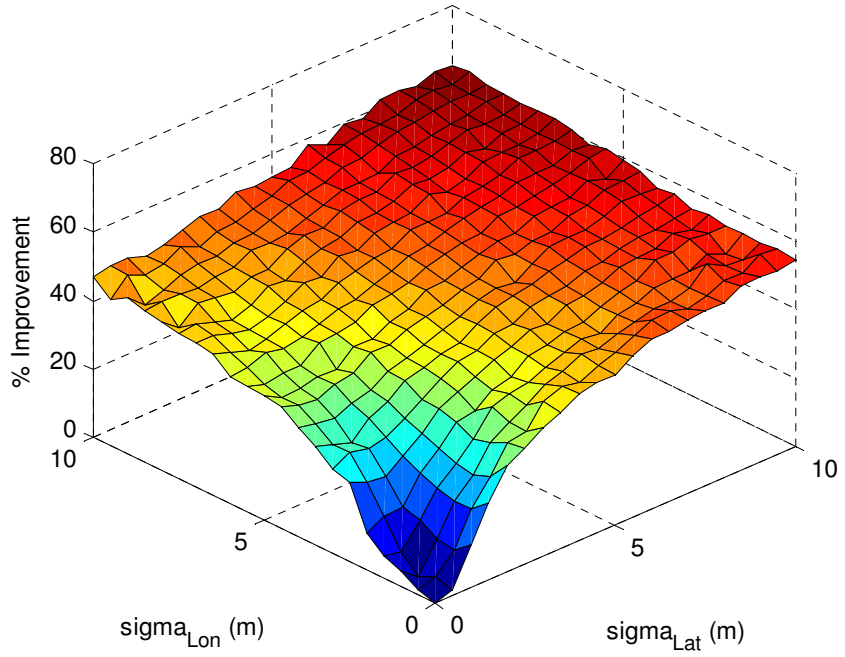


Figure 8. Percentage improvement in the accuracy of estimation after data fusion using the EKF model (at sampling rate of 1Hz) normalized with respect to the noisy data

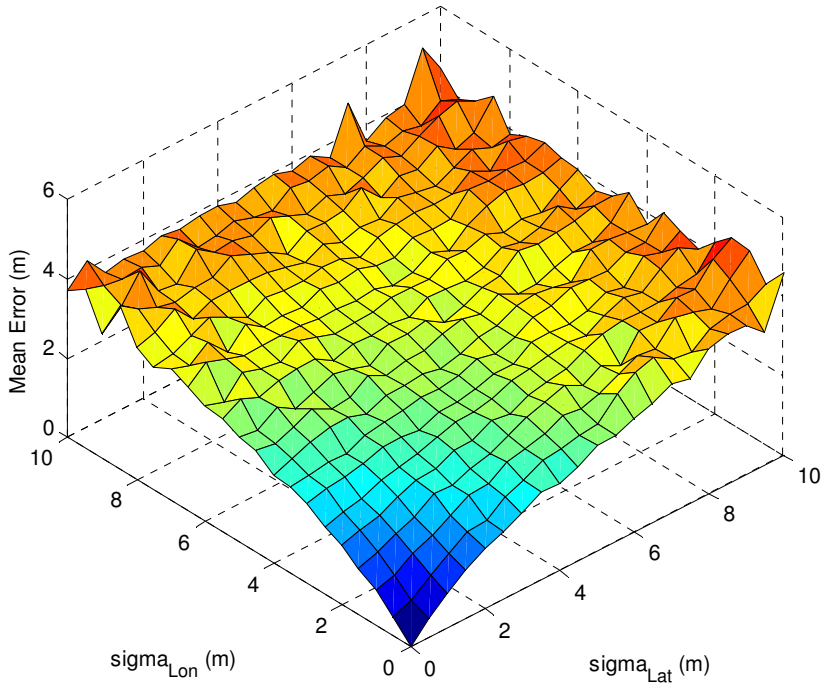


Figure 9. Mean path tracking error for vehicle (v=10m/s, Ts = 3s at different levels of noise (after data fusion).

Figure 9 shows the mean path tracking error at the sampling rate of 0.33Hz. The magnitude of the mean error as compared to the sampling rate of 1 Hz (see Figure 7) has increased. Figure 10 shows the improvement in the performance accuracy at the sampling rate of 0.33Hz. The accuracy has increased by up to 52% and the performance improvement as compared to noisy data is higher at higher levels of noise.

In general, as the amount of noise is increased, the accuracy of estimation decreases. There is a sharper decrease in accuracy as the sampling interval is increased. Considering the mean trajectory error for noisy data as the normalization factor, the percentage improvement in accuracy calculated using (2.22) is higher at higher noise levels.

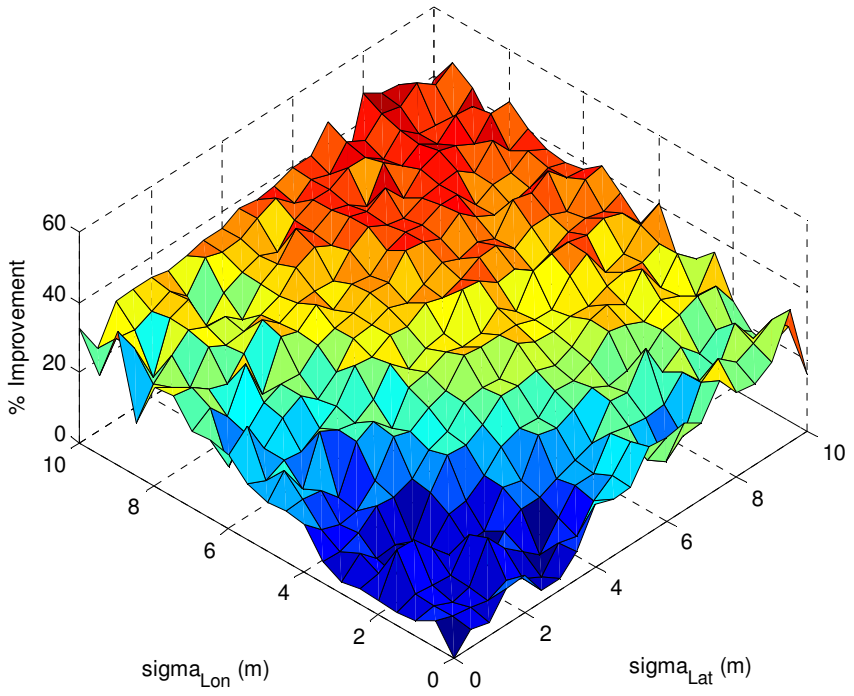


Figure 10. Percentage improvement in the accuracy of estimation after data fusion using the EKF model (at sampling rate of 0.33 Hz) normalized with respect to the noisy data

In order to analyze the effect of the scaling factor in the vehicle position estimation, data were taken at different sampling intervals ranging from 1s to 10s. The amount of noise added to the actual data was ($\sigma_{Lat,}$ σ_{Lon}) = (10, 10). Optimal S as defined in (2.18) was calculated numerically by incrementing S from 0.5 to 3 at intervals of 0.05. Table I gives a list of optimal S at different sampling intervals.

The data in Table I suggests that a scaling factor of 1 as in the conventional Kalman Filter is not always optimal. The relationship between the optimal S and the sampling intervals was not pronounced when using this analysis. Further study is necessary in order see if there is any relationship between the sampling intervals and the scaling factor.

Table I: The Optimal Values for the Scaling Factor for Different Sampling Intervals

Ts	Min $\bar{e}$	Scaling factor
1	3.2651	1.00
2	3.7523	1.05
3	3.7970	1.05
4	3.8935	1.15
5	4.0060	1.20
6	3.8877	1.00
7	3.9730	1.20
8	4.1692	1.10
9	4.2971	1.20
10	4.1909	1.25

An analysis of the relationship between road curvature and the EKF performance was carried out in the test paths shown in Figure 5. The mean error and the variance of the trajectory tracking error were calculated for the test paths at different sampling intervals.

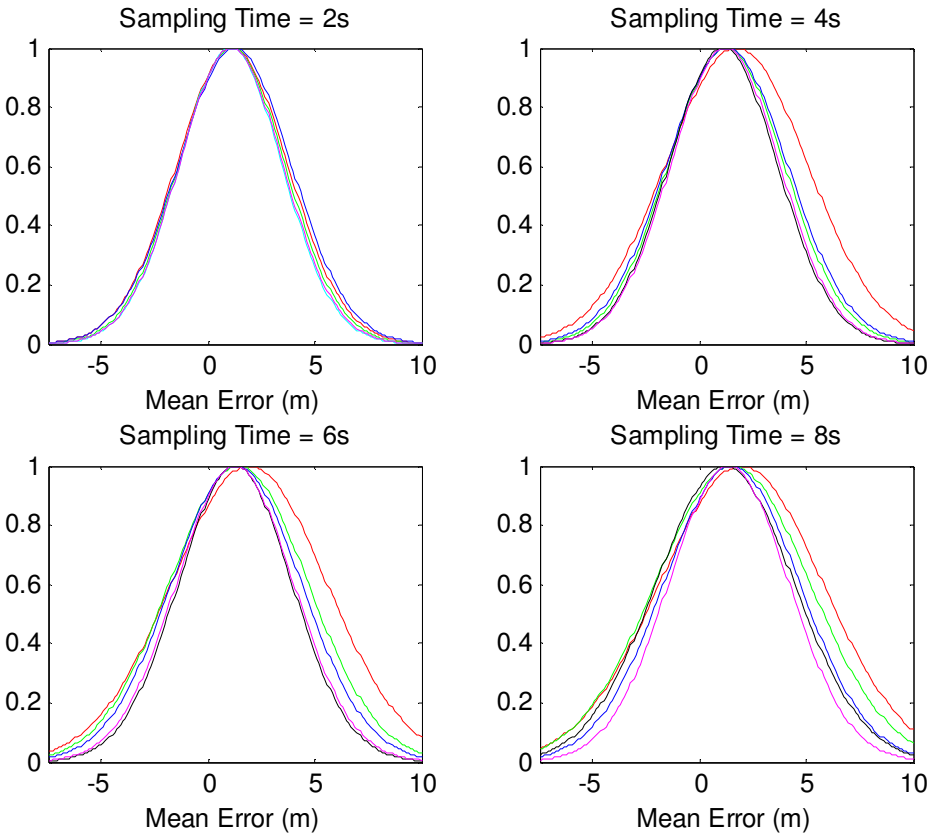


Figure 11. The distribution of mean tracking error for different curvature segments at different sampling rates

Figure 11 depicts a proportional relationship between the curvature and the performance of the data fusion model. There are larger errors in higher curvature segments of the road. There is increase in the variance and the mean as the sampling interval increases. Figure 12 shows this relationship between the increase in the sampling time and the mean tracking error at different curvatures.

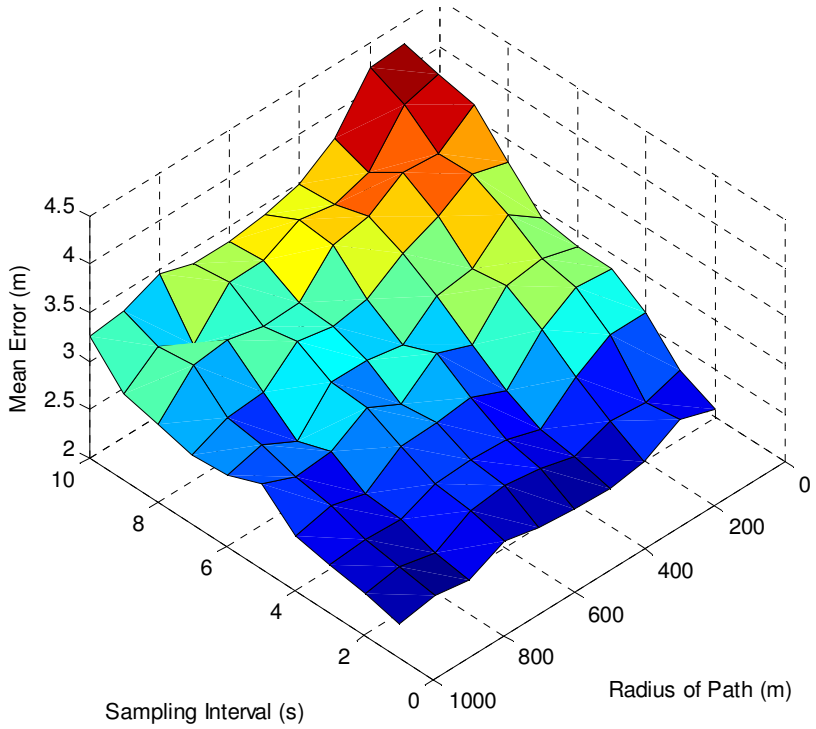


Figure 12. The distribution of mean tracking error for different curvature segments at different sampling rates

The error is proportional to the curvature and the sampling time. This is an important finding in the sense that at higher curvature roads, we must sample faster if we want to maintain the amount of error within a certain bound. R-square test were performed to verify this relationship between the sampling time, road curvature and the mean tracking error.

Table II: The R^2 Statistic For The Dependance Of Mean Error On Road Curvature At Various Sampling Times

Ts	1	2	3	4	5	6	7	8	9	10
R^2	0.38	0.55	0.7	0.75	0.8	0.88	0.93	0.8	0.9	0.9

Table II contains the results of the R^2 test on the dependence of the mean error on the curvature of the path at different sampling intervals. These values are based on a 2^{nd} order polynomial approximation of the relationship between the curvature of the road and the mean error. As the sampling intervals become higher (>4s), the dependence of the mean error on the curvature of the road becomes stronger.

Table III: The R^2 Statistic for the Dependance of Mean Error on Sampling Times at Various Curvature Roads

κ	.01	.005	.0033	.0025	.002	.0017	.0014	.0013	.0011	.001
R^2	0.97	0.98	0.91	0.91	0.91	0.96	0.96	0.92	0.95	0.95

Table III shows the results of the R^2 test on the dependence of the mean error on the sampling times at different values of curvatures. These values are based on a linear approximation of the relationship between the sampling intervals and mean error. The R2-test validates the strong dependence between the sampling times and mean error. This relationship is independent of the road curvature.

2.6 Conclusion

The study was aimed at analyzing the Kalman Filter performance at various levels of noise, sampling rates and curvatures. Through the simulation results we have shown that the GPS/INS data fusion using EKF performance in terms of the accuracy of the position estimation degrades as the amount of measurement noise increases. The EKF performance also depends on the sampling times and the curvature of the road. R^2-test on relationship

between the sampling time and the mean tracking error shows that mean error is linearly dependent on the sampling time (R^2~0.95) irrespective of the curvature of the road. However, the R^2-test on the dependence of the mean error on the curvature is weaker and suggests a quadratic relationship between curvature and mean error. The dependence on curvature is also influenced by the sampling interval in the sense that at higher sampling intervals (>5s) the relationship between curvature and the mean error is more pronounced (R^2~0.9). We also showed that the conventional Kalman Filter does not always give us the optimal gain and a scaling factor can be used to obtain better results.

These findings can be used in the determination of sampling times required for the stability of a vehicle using adaptive bandwidth allocation in intelligent transportation systems.

Acknowledgment

This research was partially supported by National Science Foundation (NSF) Grant ECS-0823952 and Federal Highway Administration (FHWA) Project under Department of Transportation. The authors would also like to thank Mr. Xiaofeng Pang for his contributions at the initial phase of the project.

References

[1] T. Bhavsar, T. Chang, J. Daniel, and M.-Y. Chow, "Synthesis of Pseudo GPS Coordinates with Real Data Image Capture for Vehicular System " in *ISIE 2009* Lisbon, Portugal, June 2009.

[2] L. Zheng and M.-Y. Chow, "Adaptive Multiple Sampling Rate Scheduling of Real-time Networked Supervisory Control System - Part I," in *IEEE Industrial Electronics, IECON 2006 - 32nd Annual Conference on*, 2006, pp. 4604-4609.

[3] L. Zheng and M.-Y. Chow, "Sampling Rate Scheduling and Digital Filter Co-design of Networked Supervisory Control System," in *Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on*, 2007, pp. 2893-2898.

[4] U. Ojha, B. R. Klingenberg, and M.-Y. Chow, "Dynamic Bandwidth Allocation Based on Curvature and Velocity in a Fleet of Unmanned Ground Vehicles," in *IROS* St Louis, MO, oct 2009.

[5] M. Wei and K. P. Schwarz, "Testing a decentralized filter for GPS/INS integration," in *Position Location and Navigation Symposium, 1990. Record. The 1990's - A Decade of Excellence in the Navigation Sciences. IEEE PLANS '90., IEEE*, 1990, pp. 429-435.

[6] J. Chaffee, J. Abel, and B. K. McQuiston, "GPS positioning, filtering, and integration," in *Aerospace and Electronics Conference, 1993. NAECON 1993., Proceedings of the IEEE 1993 National*, 1993, pp. 327-332 vol.1.

[7] W. Gao, Y. Yang, X. Cui, and S. Zhang, "Application of Adaptive Kalman Filtering Algorithm in IMU/GPS Integrated Navigation System," *Geo-Spatial Information Science,* vol. 10, pp. 22-26, March 2007.

[8] C. Hide, T. Moore, and M. Smith, "Adaptive Kalman Filtering for Low-cost INS/GPS," *The Journal of Navigation,* vol. 56, pp. 143-152, 2003.

[9] A. M. Schneider and J. L. Maida, "A Kalman filter for an integrated Doppler/GPS navigation system," in *Position Location and Navigation Symposium, 1988. Record. Navigation into the 21st Century. IEEE PLANS '88., IEEE*, 1988, pp. 408-415.

[10] M. S. Smith and G. S. Ladde, "Processing of filtered GPS data," *Aerospace and Electronic Systems, IEEE Transactions on,* vol. 25, pp. 711-728, 1989.

[11] C. Seong Yun, K. Byung Doo, C. Young Su, and C. Wan Sik, "Observability analysis of the INS/GPS navigation system on the measurements in land vehicle applications," in *Control, Automation and Systems, 2007. ICCAS '07. International Conference on*, 2007, pp. 841-846.

[12] Y. Geng and J. Wang, "Adaptive estimation of multiple fading factors in Kalman filter for navigation applications," *GPS Solutions,* vol. 12, pp. 273-279, 2008.

[13] A. Y. Kibangou and A. Monin, "GPS based land vehicle positioning using Gaussian sum filters," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, 2008, pp. 3653-3656.

[14] J. Zhu, "Conversion of Earth-centered Earth-fixed coordinates to geodetic coordinates," *Aerospace and Electronic Systems, IEEE Transactions on,* vol. 30, pp. 957-961, 1994.

Chapter 3- Dynamic Bandwidth Allocation based on Curvature and Velocity in a Fleet of Unmanned Ground Vehicles

Unnati Ojha, Bryan R. Klingenberg and Mo-Yuen Chow
Advanced Diagnosis, Automation and Control (ADAC) Laboratory
North Carolina State University,
Raleigh, North Carolina-27695
uojha,brklinge,chow@ncsu.edu

This chapter has been submitted for publication to the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2009), St. Louis, MO, Oct. 2009

3.1 Abstract

As the number of nodes in large scale network control systems increases, allocating constant bandwidth to all the nodes is neither practical nor efficient in terms of quality of control because of the limited amount of available bandwidth. In order to resolve this issue of bandwidth management in a networked control system consisting of several control loops, dynamic methods for resource allocation should be used. Recent studies have attacked this problem from the perspective of efficient bandwidth usage. In this paper, we present a novel method for dynamic bandwidth allocation in a fleet of Unmanned Ground Vehicles (UGV) that improves the quality of control while meeting the bandwidth constraints. The proposed method adjusts the amount of bandwidth allocated to each UGV according to its velocity and the curvature of the path it is tracking. Simulation results have shown significant improvement in the performance of the system.

Keywords: Networked Control Systems, Dynamic Bandwidth Allocation, Curvature, Path tracking, Intelligent Space (iSpace)

3.2 Introduction

In distributed large scale networked control systems (NCS), it is important to efficiently manage the amount of available bandwidth in order to optimize the overall performance of the system. The amount of data to be transferred through the network may be different for different agents (e.g., sensors, actuators, and controllers) in the NCS; thus requiring the need to allocate the bandwidth. The amount of bandwidth required could

depend on different factors according to the nature of the control system and the operating environment local to each agent in the system. One common example of such distributed networked control system (DNCS) is a fleet of unmanned ground vehicles (UGVs), following specific trajectories in a real-world traffic scenario, that are constantly communicating stream of sensed data over the network to a central supervisory controller. The controller then makes its decision based on all the available data from different UGVs and communicates the control signal over the network to each UGV. Other examples of distributed control systems include industrial automation, building automation, unmanned air vehicles, unmanned water vehicles etc. Each application may have different requirements for timing and sampling periods.

Different types of scheduling algorithms are used to solve the bandwidth allocation problem in networked control systems. Most of these fall under two categories: static [1, 2], and dynamic [3-9] bandwidth allocation. Static methods like round-robin [1], priority queuing [2] and First-In First-Out are not flexible and cannot adapt to the changed environment. If we consider the UGV example mentioned in the first paragraph, there might be some scenarios for example a UGV passing a narrow corner. In this case, a UGV would need to send the updates regarding its position and consequently get control signals from the controller more frequently so that it can avoid colliding against the wall. This means that the UGV would need relatively more bandwidth than others so that it can send and receive the data without any delay and loss. We would need to assign the bandwidth instantaneously in such situations so that the UGV performance is not degraded. This kind of bandwidth allocation is called dynamic bandwidth allocation. While dynamic bandwidth allocation

algorithms may make the performance better they take more computation time. However, in a networked control system, computation time of a proper designed bandwidth management system might not be substantial in compared to other delays [10].

A preemptive rate monotonic scheduling algorithm is used in [3] where the tasks with shorter periods have higher priorities. The sampling period of the NCS is determined by the congestion levels in the network. Al Hammouri et al. in [6] have considered bandwidth allocation as a control problem thus creating a second loop in the control system where controllers are designed to enhance the performance of the bandwidth allocation algorithm among several NCS loops. In [9] bandwidth allocation problem is viewed from the perspective of the computation time required and the use of distributed genetic algorithm is suggested for faster and fault tolerant bandwidth allocation. All of the papers above, however, are more concerned with the efficient use of bandwidth rather than the performance of the control loop.

There are a few papers where the bandwidth allocation problem is confronted from the controlled system's performance perspective. Large Error First (LEF) technique is used in [4] that uses feedback information from the control loop is to determine the bandwidth to be assigned to each individual component. In this paper, the plants in transient states are given the first priority followed by the control loops that have largest error. A similar study is done in [8] in a token-type network where bandwidth is allocated in accordance with the jitter. Walsh and Ye [11] have considered allocating the bandwidth based on the weighted error thus giving a higher priority to a system with higher error. Furthermore, a stability study on maximum allowable delay is done in [7] and each sampling period is divided into

real-time data cycle and non-real time data cycle. Auction based bandwidth-allocation is used in [5] where each NCS competes for the bandwidth. A utility function based on average sensitivity is used to evaluate the effects of the delays on a NCS which influences the bid that a NCS places while competing for the bandwidth. Average sensitivity is defined as the change in the output with reference to the change in the input. The average sensitivity is pre-computed and a look-up table is used in order to avoid any additional computation time delay. Li and Chow [12] have used adaptive sampling rate scheduling (ASRS) process in bandwidth allocation to maintain the fidelity of the transmitted signal.

In this paper, dynamic bandwidth allocation is based on the curvature of the path the UGV is tracking and the velocity of that UGV. This work is inspired from the natural human driving behavior. When we drive, we tend to be more cautious in curved roads rather than straight roads. In addition, our attention is also proportional to how fast we are driving our car. Analogous to our attention is the sampling rate that is required by the UGV for a better performance. This paper is different in the sense that unlike some related works [4, 5, 7, 11] where bandwidth is allocated based only on internal (UGV) status, it also considers the affect of the environment it is travelling on as one of the factors for bandwidth allocation. Furthermore, in calculating bandwidth based on curvature, we are looking ahead of time and predicting and avoiding the possible future errors rather than using present errors [4, 11] and correcting them.

The paper is organized in the following way. Section II presents the system model that we are using to implement the dynamic bandwidth allocation algorithm. In Section III we describe the algorithm we are using and formulate the problem mathematically. The test-

bed that is used for implementation of the system is described in section IV. We present the simulation results in section V and section VI concludes the paper.

3.3 System Description

The distributed network control system in this research is the path tracking control of multiple UGVs from a central supervisory controller. Each UGV is a differential drive mobile robot with two driving wheels and one or more caster wheels. The dynamics of the UGV can be described using the kinematic model in (3.1)

$$\begin{bmatrix} v_{ref} \\ \omega_{ref} \end{bmatrix} = \begin{bmatrix} \frac{\rho}{2} & \frac{\rho}{2} \\ \frac{\rho}{W} & -\frac{\rho}{W} \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}, \qquad (3.1)$$

where v_{ref} is the linear velocity of the UGV, and ω_{ref} is the angular velocity or the UGV, ρ is the radius of the drive wheels and W is the distance between the drive wheels. ω_r and ω_l variables above represent the angular velocities of the right and left drive wheel respectively.

The controller is given a path, defined by a set of consecutive waypoints on the ground, for the UGV to follow. Given the UGVs position the controller generates a desired reference velocity and turn rate so that the path will be tracked. With a specified control signal the inverse of the kinematics in (3.1) can be used to calculate the reference speeds for each wheel in order to achieve the linear velocity and turn rate. The UGV implements a PI controller to provide the desired ω_r and ω_l. A differential drive robot with fixed reference speeds for each wheel will travel in a arc until it receives new reference speeds.

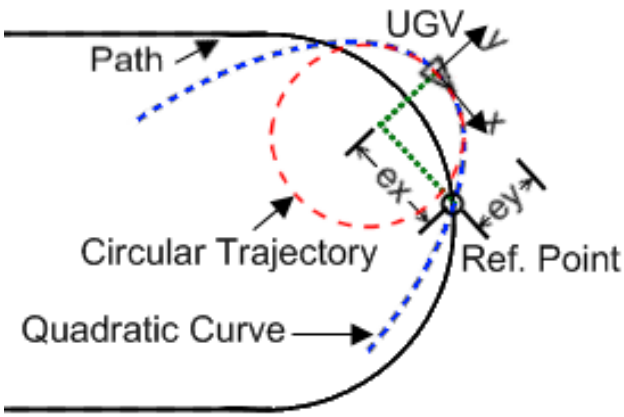


Figure 13: Quadratic Curve path tracking

To calculate the control signal for the UGV the controller implements the quadratic curve path tracking algorithm described in (3.2)-(3.5). In this algorithm a reference point, located at some look-ahead distance, is moved along the predefined path. As the reference point moves along the path and the UGV tracks the reference point the UGV tracks the path. To track the reference point the controller finds a quadratic curve that passes through the reference point and has its vertex located at the UGV. A velocity and turn rate can then be calculated such that the UGV will move along the quadratic curve. As shown in [13] the velocity and turn rate can be simplified to the values shown below since the sampling period is sufficiently small.

$$A = \frac{e_x}{e_y^2}, \qquad (3.2)$$

$$K = \mathrm{sgn}(e_x)\frac{\alpha}{1+|A|}, \qquad (3.3)$$

$$v_{ref} = K, \qquad (3.4)$$

$$\omega_{ref} = 2AK, \qquad (3.5)$$

where K is inversely proportional to the quadratic coefficient A. This is accomplished by replacing the quadratic curve with a circle centered at the focus of the parabola and a radius defined by the distance from the focus to the vertex.

The curvature for the command value is then the inverse of the radius of the circle that the UGV is being command to follow. When the UGV tracking performance is poor the UGV will move away from the desired path. As shown in Figure 13, this causes the quadratic curve to be sharper thus decreasing the radius of the circular path. This decrease in radius causes the curvature of the path to be taken to increase. Since K is inversely proportional to A and from (3.4) one can see that the UGV velocity will increase when curvature, which is dependent on the reference point location, decreases.

3.4 Bandwidth Allocation Based on Curvature and Velocity

In this work, we are allocating bandwidth based on the curvature of the path and the velocity of the UGV. The calculation of the curvature (A) and velocity (v) is described in section II. Consider two UGVs, A and B operating in the environment where A is moving along an almost straight line and B is right around a turning.

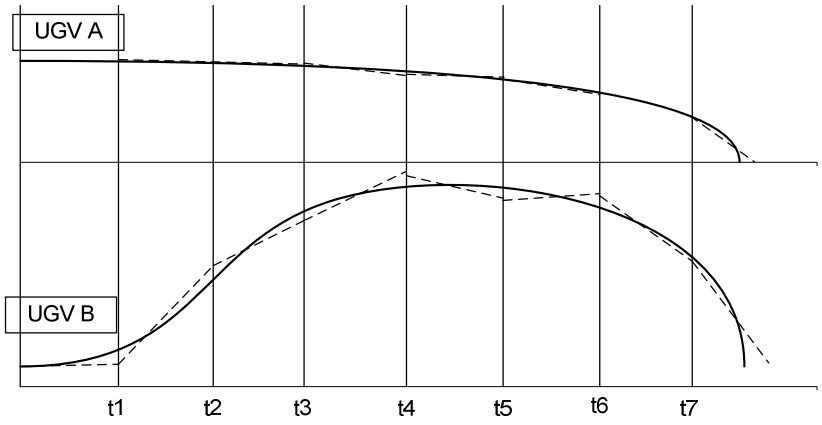


Figure 14: Path tracking using the same sampling rate for different curvature path

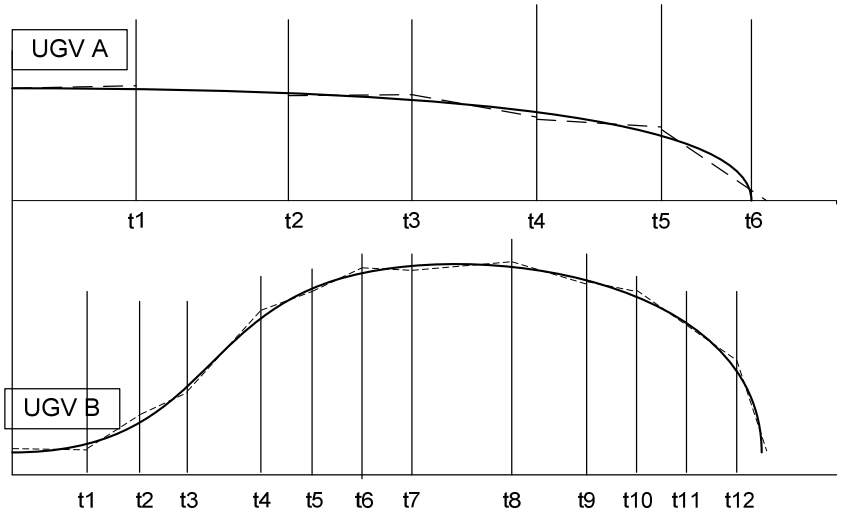


Figure 15: Path tracking using different sampling rates based on curvature

In Figure 14, if A and B get the same amount of bandwidth, and thus the same sampling rate, A's position and control signal will be updated more often than necessary. On the other hand, B's position and control signal will be updated less number of times than necessary to traverse smoothly around the curve. Thus for the trajectory of B we get a very

rough estimation of the curve (the dashed line represents the path tracked by the UGV). However, in Figure 15, where UGV B now gets a greater portion of the bandwidth than A, thus having a higher sampling rate, the curve tracking for B is much smoother without actually considerably affecting the tracking of A. We can intuitively say that the amount of tracking error is lesser in case II (Figure 15) then the former case. Based on this intuition, we can say that the allocated bandwidth should be proportional to the curvature of the road.

Similarly, assuming that UGV A and B are travelling on a same curve but A is moving at half the speed of B. Consider case I where A and B have same amount of bandwidth allocated. At the end of each sampling period, the amount of overshoot would be greater in case of B than in case of A. However, once again giving more bandwidth to B would mean shorter sampling periods for B. This in turn would limit the amount of overshoot occurring in B. This reduction in overshoot is greater than the amount of overshoot added in A because of its greater sampling period. Thus we intuitively propose that the amount of bandwidth allocated should be proportional to the velocity of the UGV.

Based on the two propositions above, bandwidth for each UGV was calculated using(3.6).

$$BW_i = \left(w_1 \frac{A_i}{\sum_j A_j} + w_2 \frac{v_i}{\sum_j v_j} \right) BW_{total} . \qquad (3.6)$$

where, BW_i is the portion of bandwidth allocated for UGV *i*. A_i is the instantaneous curvature of the path that UGV *i* is travelling on. v_i is the instantaneous velocity of UGV *i*. w_1 and w_2 are the weights given to the curvature and velocity respectively. BW_{total} is the total

available bandwidth. i and j both range from 1 to N, where N is the number of UGVs (N=4 in our case).

The above equation satisfies the bandwidth constraint

$$\sum_i BW_i \le BW_{total}, \qquad (3.7)$$

as long as we have

$$w_1 + w_2 = 1. \qquad (3.8)$$

After the amount of bandwidth to be allocated to each UGV is calculated, the sampling period is calculated as

$$\Delta t_i = \frac{2}{BW_i}, \qquad (3.9)$$

where Δt_i is the calculated sampling time for UGV i. The sampling trigger tr is a integral factor of the minimum sampling time and is defined as:

$$tr_i = \left\lceil \frac{\Delta t_i}{\Delta t_{\min}} \right\rceil, \qquad (3.10)$$

where $\Delta t_{\min}$ is the minimum possible sampling time for the system. The sign $\lceil a \rceil$ represents the largest integer greater than or equal to a. Thus, for UGV i the sensor data and control signals are communicated at every tr_i sampling periods. The value for tr is updated after every $2tr_{\max}$ periods.

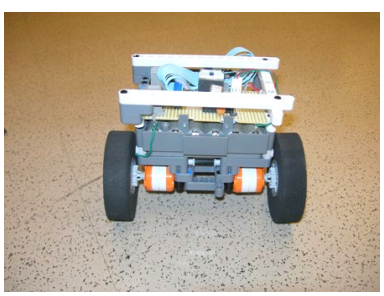
3.5 Test-bed

A network based integrated navigation system has different modules combined together to guide a UGV from one point to another in the space of interest where the

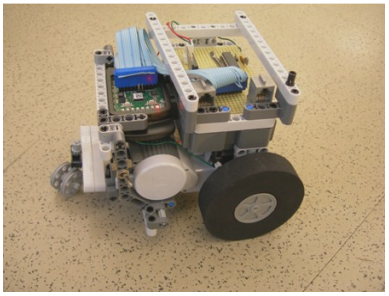
navigational intelligence comes from a remote controller. The different modules in the space of interest can be combined together using the intelligent space concept. The intelligent Space (iSpace) concept has been used to effectively implement distributed sensors, distributed actuators (e.g. robots), distributed processors, and information technology over physically connected space [14] and spaces connected via communication networks[15].

The intelligent space implemented in this project has a network of distributed sensors (cameras and encoders) and multiple actuators (UGVs). iSpace aggregates the entire space status from each agent's sensory information and responds intelligently to the system goals. The mobile agents in this implementation are UGVs that are controlled by iSpace to move between locations tracking unique paths to avoid obstacles. The vision system is used to locate all of the objects within iSpace so that the user can select individual destinations for each robot. Once destinations are selected the controller uses the data available to generate unique paths for each UGV using the fast marching method described in [16].

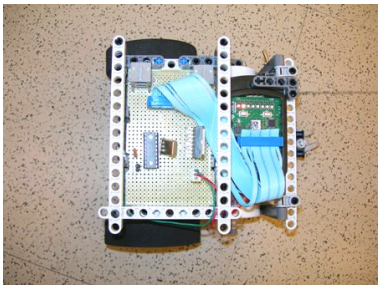
The UGVs are differential drive mobile robots with one caster wheel. The UGVs have been built using the LEGO mindstorms NXT TriBot as a base but the controller has been replaced with a SPOT controller from Sun Microsystems. The UGV used in this project, constructed using the LEGO TriBot and the Sun SPOT, has been called a TriSPOT as shown in Figure 16. The TriSPOT simulation model is composed of the vehicle kinematics and dynamics, the motor dynamics and the motor speed controllers.



Front View



Side View



Top View

Figure 16: A TriSPOT.

The TriSPOTs communicate with the supervisory controller using the SPOT's IEEE 802.15.4 compliant radio. The delays in this radio signal are different than the delays in Ethernet communications and so the simulation model represents this. While all of the TriSPOTs execute within the same intelligent environment they each have their own independent tasks. Each TriSPOT has a path that it tracks using the methods described in the system setup section. The controller generates control signals in parallel for each TriSPOT and transmits the signals using the wireless radio.

3.6 Results from Simulation

Matlab/Simulink was used to evaluate the bandwidth allocation algorithm using four TriSPOTs with trajectories that had four different curvatures. Equations (3.2)-(3.5) were used

to calculate the curvature and velocity and Equations (3.6)-(3.10) were used to calculate the sampling times.

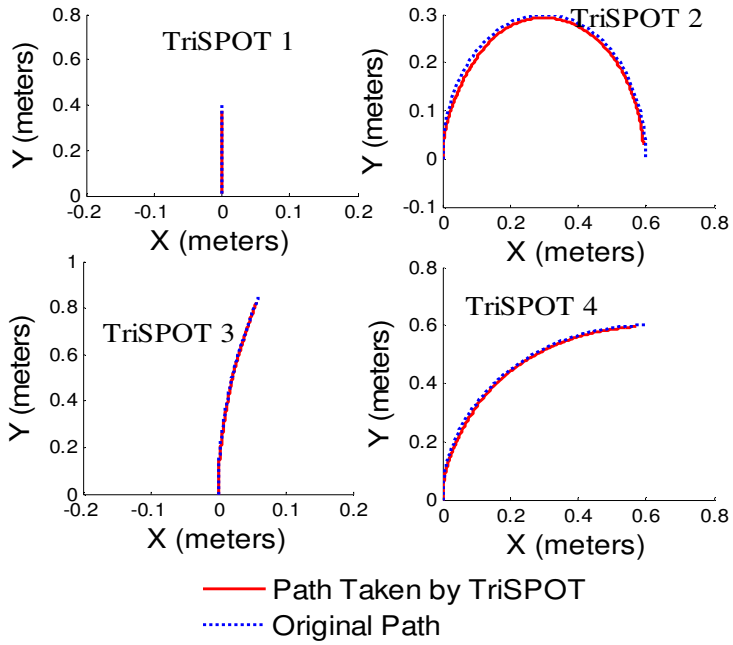


Figure 17: The paths set up for testing the bandwidth allocation algorithm

Figure 17 shows the paths for the four TriSPOTs. The first path is a straight line; the second path has a radius of 0.3m; the third path has a radius of 6m and the fourth path has a radius of 0.6m. The paths are so chosen to demonstrate the effect of different curvature paths in the bandwidth allocation algorithm.

If our algorithm were just based on curvature, the first path should get the least amount of bandwidth and the second path should get the highest amount. However, because the bandwidth depends both on velocity and curvature, the distribution of bandwidth does not behave as such.

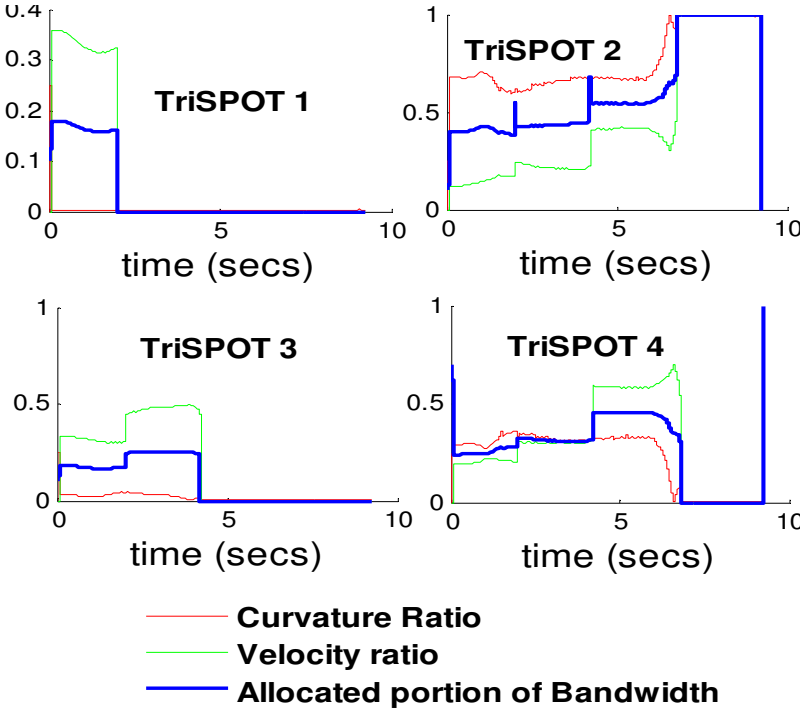


Figure 18: Bandwidth Allocation for each TriSPOT based on curvature and velocity

A plot of the ratio of the curvatures and the velocity for each TriSPOT is given in Figure 18. The portion of the bandwidth allocated for each TriSPOT is the average of the two ratios. The first TriSPOT, in spite of being in a straight line gets slightly more bandwidth than TriSPOT 3 because of its high velocity. As the first TriSPOT reaches its destination at around time t = 2s, the amount of bandwidth is now distributed only among the three remaining TriSPOTs. After each TriSPOT reaches the destination, we can observe an increase/jump in the portion of bandwidth allocated to the remaining TriSPOTs. The total amount of bandwidth usage at any point in time is equal to the total available bandwidth.The algorithm was evaluated based on the total accumulated error. The total accumulated error was calculated using (3.11) - (3.12).

$$e_i = \sum_j \min\left(\left\|\vec{x}_{i,j} - \vec{x}_{i,p}\right\|\right) \times \Delta t\,, \qquad (3.11)$$

where, $x_{i,\,j}$ is a 2×1 vector representing the position of the TriSPOT *i* at instance *j*. $x_{i,\,p}$ is an (2×*p*) matrix of discrete path points that TriSPOT *i* was intended to follow. Δt is the sampling time. *j* ranges from 1,..,*p*. *i* represents the number of TriSPOTs and ranges from 1..4. *p* is the number of waypoints in the path. The value of *p* varies according to the length of the path the TriSPOT is tracking.

$$e_{total} = \sum e_i\,, \qquad (3.12)$$

where e_i represents the total accumulated error for TriSPOT *i* and e_{total} is the total accumulated error for the system.

Figure 19 shows the accumulated error for each of the four TriSPOTs when the system was run without any bandwidth allocation (Left) and when system was run by implementing dynamic bandwidth allocation using the proposed algorithm (Right). The total bandwidth used for this simulation was 20bps. Additional simulations were done to verify the results obtained by varying the total available bandwidth.

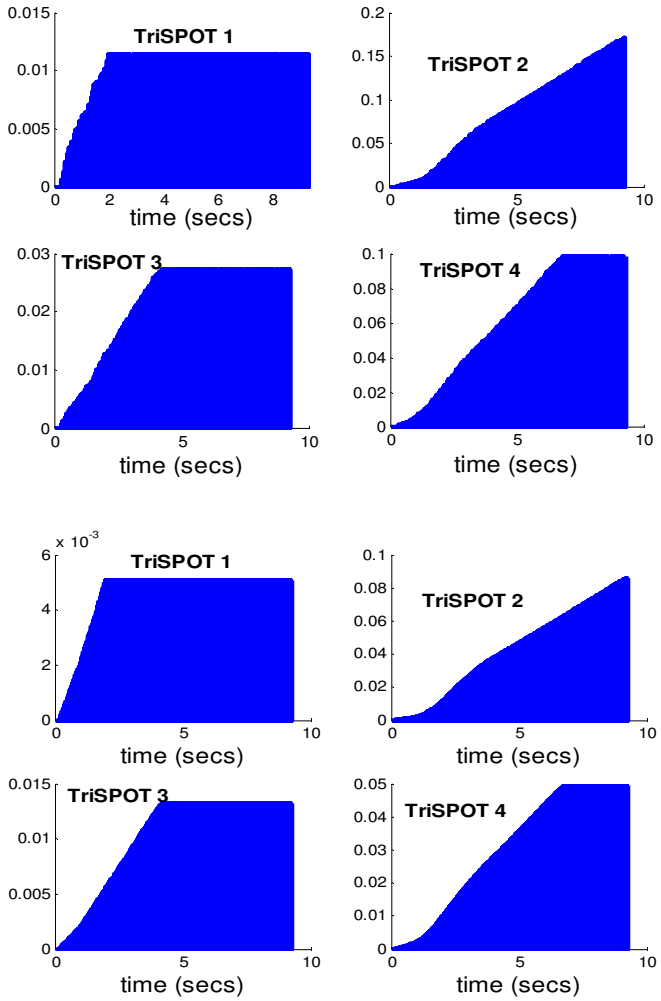


Figure 19: Total Accumulated error for each TriSPOTs when using constant bandwidth allocation (top) and dynamic bandwidth allocation (bottom)

The results indicate that the performance of the system has increased considerably based on the reduction in the accumulated error. In this simulation, the accumulated error was reduced by 55.31%, 49.56%, 51.65% and 49.80% for TriSPOT 1, TriSPOT 2, TriSPOT 3 and TriSPOT 4 respectively. The total error was reduced by 50.03%. Table lists down the total accumulated errors and the percent reduction error in each case.

Table IV: Total accumulated error for each TriSPOT and the overall reduction in the error for the system

Bandwidth		10bps	20bps	30bps
TriSPOT 1	Const BW	0.0154	0.0113	0.0097
	Var BW	0.0049	0.0051	0.0043
TriSPOT 2	Const BW	0.274	0.1707	0.1375
	Var BW	0.0867	0.0861	0.0675
TriSPOT 3	Const BW	0.0428	0.0273	0.0234
	Var BW	0.0133	0.0132	0.0108
TriSPOT 4	Const BW	0.1604	0.0984	0.0785
	Var BW	0.0496	0.0494	0.039
Total	Const BW	0.4926	0.3077	0.2492
	Var BW	0.1545	0.1538	0.1216
	% Reduction	**68.64**	**50.03**	**51.18**

3.7 Conclusion

In this paper, we have presented a novel bandwidth allocation algorithm in a networked control system based on curvature and velocity of the UGV. This new algorithm is based on how human beings focus their attention while driving their cars. The results show that there is a considerable amount of increase in the performance of the system when using the proposed algorithm. On implementing this algorithm, there was 50-60% reduction in the accumulated tracking error.

Acknowledgment

This research was supported under NSF-ECS- 0823952 “Impaired Driver Electronic Assistant (IDEA)” project and Department of Transportation Massive Sensor Based Congestion Management System for Transportation System project.

References

1. Chaskar, H.M. and U. Madhow, *Fair scheduling with tunable latency: a round-robin approach.* Networking, IEEE/ACM Transactions on, 2003. **11**(4): p. 592-601.

2. Forouzan, B.A., *Data Communications and Networking*. 3 ed. 2004, Singapore: McGraw Hill.

3. Brainicky, M.S., Philips, S M & Zhang, W. , *Scheduling and Feedback Co-Design for Networked Control Systems*, in *IEEE/CDC*. 2002: Las Vegas, USA.

4. Yepez, J., P. Marti, and J.M. Fuertes. *Control loop scheduling paradigm in distributed control systems*. in *Industrial Electronics Society, 2003. IECON '03. The 29th Annual Conference of the IEEE*. 2003.

5. Tipsuwan, Y., et al., *An auction-based dynamic bandwidth allocation with sensitivity in a wireless networked control system.* Computers & Industrial Engineering, 2008.

6. Al-Hammouri, A.T., et al. *Decentralized and dynamic bandwidth allocation in networked control systems*. in *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*. 2006.

7. Hong Seong, P., et al., *A scheduling method for network-based control systems.* Control Systems Technology, IEEE Transactions on, 2002. **10**(3): p. 318-330.

8. Tao, B., W. Zhi-ming, and Y. Gen-ke, *Optimal bandwidth scheduling of networked control systems (NCSs) in accordance with jitter.* Journal of Zhejiang University - Science A, 2005. **6**(6): p. 535-542.

9. Hidehiro, K., et al., *Designing a distributed algorithm for bandwidth allocation with a genetic algorithm*. 2004, Wiley-Interscience. p. 37-45.

10. Mo-Yuen, C. and T. Yodyium. *Network-based control systems: a tutorial*. in *Industrial Electronics Society, 2001. IECON '01. The 27th Annual Conference of the IEEE*. 2001.

11. Walsh, G.C. and Y. Hong, *Scheduling of networked control systems.* Control Systems Magazine, IEEE, 2001. **21**(1): p. 57-65.

12. Zheng, L. and M.-Y.Chow. *Adaptive Multiple Sampling Rate Scheduling of Real-time Networked Supervisory Control System - Part I*. in *IEEE Industrial Electronics, IECON 2006 - 32nd Annual Conference on*. 2006.

13. Yoshizawa, K., et al. *Path tracking control of mobile robots using a quadratic curve*. in *Intelligent Vehicles Symposium, 1996., Proceedings of the 1996 IEEE*. 1996.

14. Joo-Ho, L., et al. *Adaptive guidance for mobile robots in intelligent infrastructure*. in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*. 2001.

15. Vanijjirattikhan, R., et al. *Mobile Agent Gain Scheduler Control in Inter-Continental Intelligent Space*. in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. 2005.

16. Sethian, J.A., *A fast marching level set method for monotonically advancing fronts.* PNAS, 1996. **93**(4): p. 1591-1595.

Chapter 4- Predictive Control of Multiple Robots in a Networked Control System with Adaptive Bandwidth Allocation

Bryan R. Klingenberg, Unnati Ojha, Mo-Yuen Chow
Advanced Diagnosis Automation and Control Lab
Department of Electrical and Computer Engineering
North Carolina State University,
Raleigh NC 27695, USA

This chapter has been submitted for publication to the 35th Annual Conference of the IEEE

Industrial Electronics Society (IECON '09), Porto, Portugal, November, 2009

4.1 Abstract

In network based path tracking control of systems with multiple UGVs performance can be effected by network constraints including bandwidth limitation and time-varying network delays. Network delay has previously been compensated for using smith predictor based techniques and gain scheduling to limit UGV motion. The predictive gain scheduler introduced uses the predicted trajectory and future path to maximize allowable travel while considering the network delay. On the other hand, bandwidth management has been approached by optimizing total bandwidth usage under performance constraints. However, by optimizing the performance of the UGVs under bandwidth constraints, the total networked control system performance increases. Both of these methods take into account the trajectory curvature and network conditions

4.2 Introduction

Many UGVs today communicate wirelessly with a remote controller that makes control decisions and shares global data with other controllers. These networked control systems are susceptible to delay which can adversely affect the performance of the UGV. This problem becomes especially prominent in a multi-robot system where multiple robots are sharing the same communication channel.

There are many approaches to path tracking control of differential drive UGVs such as pure pursuit [1] and quadratic curve path tracking [2] which track reference points on the path. These path tracking algorithms are effective for non-holonomic locally controlled

robots. Predictive control has been used in [3] where the predicted trajectory is compared to the future path. Others have used predictive control to compensate for effects of wheel slippage, terrain variations and curvature [4]. In [5] the curvature ahead of the vehicle is detected to limit vehicle speed such that lateral motion will be limited. Glaser et al. tackle a similar problem where long range road data is used to prevent drivers from taking turns at unsafe speeds [6].

The effect of network delay has also been studied. In [7] a feedback preprocessor, similar to a smith predictor, is presented. An elaboration on this approach is found in [8] where gain scheduling is added to prevent a control signal from causing path deviation based on network delay and trajectory curvature.

In this work the path tracking controller compensates for time varying network delay by predicting the future position of the UGV. The prediction is used, along with the constraint of the reference path, to determine how far the UGV will accurately track the path given a current state and control signal and then scale the control signal to prevent future path deviation. Since this gain scheduling result is an aggregation of several pertinent factors it is also used to determine bandwidth requirements. If a control value is determined to be effective for a large section of the future path then bandwidth will be allocated away from that UGV.

The network and CPU constraints may cause the quality of control to degrade to an extent that it may cause instability [9]. Feng Xia [9] in the same work has also suggested that bandwidth allocation can be an effective method to improve performance of control loops under such conditions. Different types of scheduling algorithms are used to solve the

bandwidth allocation problem in networked control systems. Most of these fall under two categories static and dynamic bandwidth allocation. Static methods, like round-robin [10] and priority queuing [11, 12], are not flexible and cannot adapt to the changing environment. To overcome that, adaptive bandwidth allocation techniques have been introduced. A preemptive rate monotonic scheduling algorithm is used in [13] where tasks with shorter periods have higher priorities. The sampling period of the NCS is determined by the congestion levels in the network.

There are a few papers where the bandwidth allocation problem is confronted from the controlled system's performance perspective. Large Error First (LEF) technique, used in [14], uses feedback information from the control loop is to determine the bandwidth to be assigned to each individual component. In this paper, the plants in transient states are given the first priority followed by the control loops that have largest error. A similar study is done in [15] in a token-type network where bandwidth is allocated in accordance with the jitter. Walsh and Ye [16] have considered allocating the bandwidth based on the weighted error thus giving a higher priority to a system with higher error. Zheng and Chow have proposed an adaptive sampling rate scheduling (ASRS) methodology in [17] to maintain the transmitted measurement signals' fidelity, as well as conserve available bandwidth in order to help supervisory control system under normal and abnormal operating conditions.

The remaining sections are organized as follows: Section II provides the system description of the UGV, controller and network delay. Section III introduces the feedback pre-processing and predictive constrained gain scheduler. Section IV introduces the delay

tolerant dynamic bandwidth allocation methods used. Finally, the simulation results are presented in section V.

4.3 System Description

i) Unmanned Ground Vehicle

The distributed network control system in this research is the path tracking control of multiple UGVs from a central supervisory controller over a network. Each UGV is a differential drive UGV with two driving wheels and one caster wheel. The dynamics of the UGV can be described using the kinematic model in (4.1)

$$\begin{bmatrix} v_{ref} \\ \omega_{ref} \end{bmatrix} = \begin{bmatrix} \frac{\rho}{2} & \frac{\rho}{2} \\ \frac{\rho}{W} & -\frac{\rho}{W} \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}, \qquad (4.1)$$

where v_{ref} is the linear velocity of the UGV, and ω_{ref} is the angular velocity or the UGV, ρ is the radius of the drive wheels and W is the distance between the drive wheels. The ω_r and ω_l variables above represent the angular velocities of the right and left drive wheel respectively. The UGV steers by driving the wheels at different speeds determined by solving (4.1) where v_{ref} and ω_{ref} are given as an input reference command $\boldsymbol{u} = [v_{ref} \quad \omega_{ref}]^T$. These two control parameters are used to control the three UGV states defined by (4.2)

$$\mathbf{a}(t_i) = \begin{bmatrix} x(t_i) & y(t_i) & \phi(t_i) \end{bmatrix}, \qquad (4.2)$$

where $x(t_i)$ and $y(t_i)$ are the UGV coordinates in the world frame and $\phi(t_i)$ is the UGV heading expressed as the angle between the UGV and the +x axis of the world frame.

ii) Path Tracking Controller

The controller calculates a path for each UGV in the multi-robot system to follow. The path, $\boldsymbol{p}$, is defined by a set of consecutive waypoints on the ground for the UGV to follow as $\boldsymbol{p} = \begin{bmatrix} \boldsymbol{p}_x & \boldsymbol{p}_y \end{bmatrix}$ where $\boldsymbol{p}_x$ is the set of x coordinates in the world frame and $\boldsymbol{p}_y$ is the set of y coordinates. In order to follow this path the controller implements the quadratic curve path tracking algorithm. This algorithm includes a method for determining a variable look-ahead distance based on the previous UGV control signal as well as a variable speed. A reference point ahead of the UGV is constrained to move along $\boldsymbol{p}$. As this point moves along the path and the UGV tracks it the UGV effectively tracks the path. The location of the reference point is determined by finding the nearest point on the path to the UGV and looking ahead in the path a variable distance defined using methods in [2].

To track the reference point the controller finds a quadratic curve that passes through the reference point and has its vertex located at the UGV. A velocity (4.4) and turn rate (4.5) can then be calculated by fitting a circle in the quadratic curve since the sample period is sufficiently small [2].

$$A = \frac{e_y}{e_x^2}, \tag{4.3}$$

$$v_{ref} = \mathrm{sgn}(e_x)\frac{\alpha}{1+|A|}, \tag{4.4}$$

$$\omega_{ref} = 2Av_{ref}. \tag{4.5}$$

where A is the quadratic curve coefficient, α is the maximum velocity. Using A to determine the velocity causes the UGV to slow down while turning to prevent error. This is opposed to pure pursuit which simply fits a circle between the UGV location and a reference point on the path at some fixed look-ahead distance and uses a constant velocity [1]. The curvature for the command value is then the inverse of the radius of the circle that the UGV is being command to follow. This curvature can also be found from the value of ω_{ref} and v_{ref} as:

$$A = \frac{\omega_{ref}}{2v_{ref}} . \qquad (4.6)$$

Since the reference command is a circular trajectory defined by (4.4) and (4.5) it is required that the controller frequently generates new quadratic curves and subsequent circular arcs for the UGV. The UGV will follow the reference circular trajectory for a short period of time such that when all of the circular arcs are aggregated together they approximate the quadratic curve which in turn approximates the desired path.

iii) Multi-Robot Networked Control System

This is a multi robot system that is operating in a networked control environment. Each UGV sends position feedback in the form of encoder counts to a central controller. The robots are all sharing one communication link back to the supervisory controller for encoder feedback and control signals. The supervisory controller manages individual control loops for each UGV in the system maintaining separate paths, settings, and model parameters and state for the individual UGVs.

iv) Network Delay

The UGVs in consideration are controlled over a wireless network that causes the control signals and feedback signals to be delayed. This delay can be represented as the time that a signal takes to be transmitted from the controller to the robot, τ_{CR}, and the time that a signal takes to be transmitted from the robot to the controller, τ_{RC} 0. The sum of these delay times is the round trip delay, $\tau_{RTT} = \tau_{CR} + \tau_{RC}$. The nature of the delay is dependent on the type of network used for communication [18].

4.4 Predictive Delay Compensation

i) Feedback Pre-processing

In this work we use a feedback pre-processor (FP) to predict the position of the UGV when the control value will be executed. The pertinent future position of the UGV is the position at $t = t_i + \tau_{CR}(i)$ where t_i is the time when the control signal is received by the UGV. The position predicted by the FP is used by the path tracking controller and can be defined as $\hat{\mathbf{a}}(t_i + \tau_{CR})$. The position input to the FP, $\mathbf{a}(t_i)$, however, is the position of the UGV sampled before transmission to the controller at $t = t_i - \tau_{RC}(i)$.

Given that the input to the FP is $\mathbf{a}(t_i - \tau_{RC})$ and the desired output is $\hat{\mathbf{a}}(t_i + \tau_{CR})$ the UGV movement during this period can be approximated using the previous control value $\mathbf{u}(t_{i-1}) = [v_{ref}(t_{i-1}) \quad \omega_{ref}(t_{i-1})]^T$. Using the system dynamics given in (4.1) the future position is calculated by adding the change in position during

$[t_i - \tau_{RC}, t_i + \tau_{CR}]$, this time can be replaced with $\tau = \tau_{RC} + \tau_{CR}$. Then, with position starting at

$$\mathbf{a}(t_i - \tau_{RC}) = \begin{bmatrix} x(t_i - \tau_{RC}) & y(t_i - \tau_{RC}) & \phi(t_i - \tau_{RC}) \end{bmatrix}, \quad (4.7)$$

and by estimating that $\phi = \phi(t_i - \tau_{RC})$ the change in position over τ is:

$$\Delta_\tau \phi \cong \omega_{ref}(t_{i-1})\tau, \quad (4.8)$$

$$\Delta_\tau x \cong v_{ref}(t_{i-1})\tau[\cos(\Delta_\tau \phi)\cos(\phi) - \sin(\Delta_\tau \phi)\sin(\phi)], \quad (4.9)$$

$$\Delta_\tau y \cong v_{ref}(t_{i-1})\tau[\cos(\Delta_\tau \phi)\sin(\phi) + \sin(\Delta_\tau \phi)\cos(\phi)], \quad (4.10)$$

and the predicted position $\hat{\mathbf{a}}(t_i + \tau_{CR})$ is:

$$\begin{aligned} \hat{\mathbf{a}}(t_i + \tau_{CR}) = [&x(t_i - \tau_{RC}) + \Delta x \\ &y(t_i - \tau_{RC}) + \Delta y \quad \phi(t_i - \tau_{RC}) + \Delta\phi] \end{aligned} \quad (4.11)$$

This value, $\hat{\mathbf{a}}(t_i + \tau_{CR})$, is forwarded to the controller for path tracking so that the control values will be appropriate when they arrive and are executed by the UGV. More discussion on FP can be found in [7].

ii) Predictive Constrained Gain Scheduling

A predictive constrained gain scheduler, based on [8], is implemented on the control signal to adjust the control signal according to network conditions, trajectory curvature, and allowable state change to prevent the UGV from deviating from the path. The control signal is scaled by a gain that will limit UGV motion to be less than some value, ε, that is determined dynamically at runtime based on the UGVs predicted position in relation to the future path. Predictive constrained gain scheduling allows ε to adapt until the point that the UGV actually begins to deviate

from the path. The controller does this using information about the path ahead of the robot from either the supervisory controller or onboard sensors, as well as from information about the network conditions and the next control value $\boldsymbol{u}(i+1)$.

iii) Dynamically Calculating Epsilon

To dynamically find an optimal value for ε based on the current UGV position and the future path the future position of the UGV is predicted using the next control value and future path as a constraint. The method presented in Figure 20 details how this can be accomplished.

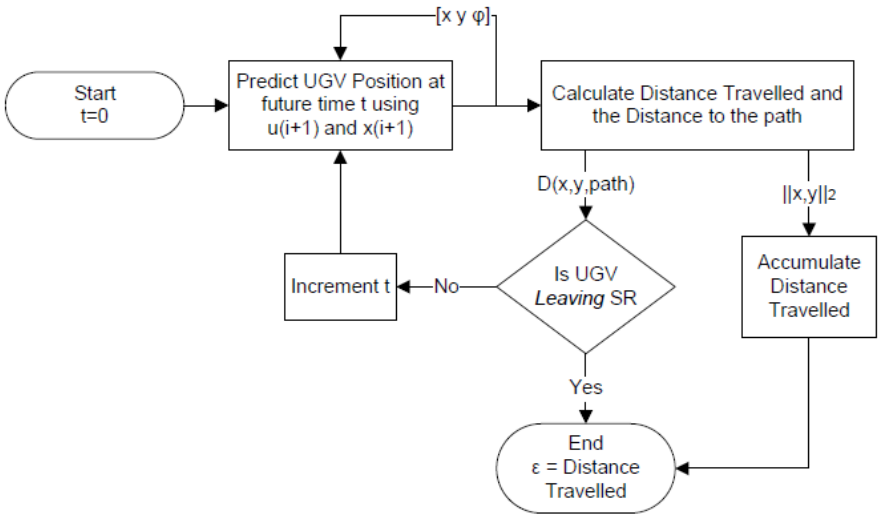


Figure 20: Algorithm for finding ε value to use

The control signal, the current state, and the UGV model are used to predict the UGV position incrementally until the UGV exits a safety region defined around the path. The time is incremented from the current time t_i by $j\Delta\tau$ where j is the step number and $\Delta\tau$ is the step resolution. The point where the UGV exits the safety region represents the point where the control signal is no longer effective at tracking

the path. Choosing ε to be the distance travelled to reach the exit point will allow the UGV to travel farther in one time step provided that the travel follows the path.

$$\varepsilon = \sum_{j=1}^{n} D(\hat{\boldsymbol{x}}_u(j), \hat{\boldsymbol{x}}_u(j-1)), \tag{4.12}$$

$$\hat{\mathbf{a}}_u(j) = \mathbf{a}_u(t_i + j\Delta\tau). \tag{4.13}$$

As shown in Figure 20, the ε value can be calculated using (4.12) where n is the point where the UGV leaves the safety region. The future position is predicted using (4.8)-(4.11) in a similar fashion to feedback pre-processing except that the value of τ is replaced with $j\Delta\tau$.

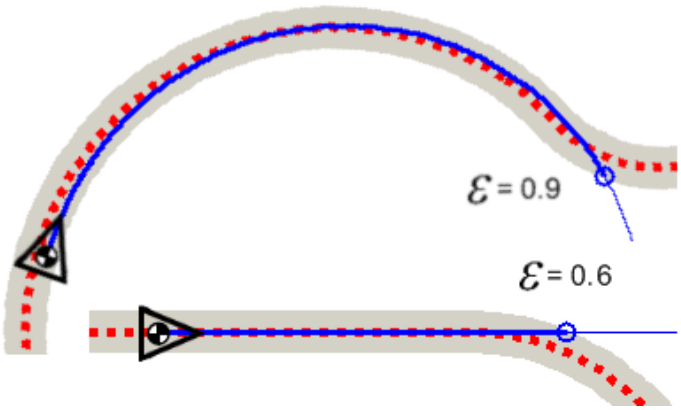


Figure 21: Predicted UGV travel and path safety region showing the effect of different epsilon values

For example, in Figure 21 two UGVs are shown travelling on two different paths. The UGV on the bottom is travelling in a straight line with a future curve. With the safety region defined around the path the UGV's position is predicted iteratively until it exits the safety region at ε=0.9. The UGV will be able to travel on the same control value without frequent updates until it begins to approach an actual

travelled distance of ε. The UGV on top in Figure 21 is tracking a curved path with curvature A. Since, the controller generates a control signal that has a trajectory that closely matches A the UGV would technically be able to travel through the entire curve without updates if the UGV model is perfect and if the system is free of noise and disturbances.

iv) Constrained Optimization of Control Values

With the allowable travel known it is possible to further evaluate the current control signal and determine how much gain scheduling is needed. To do this the following functions are used in the optimization:

$$g(i+1) = \sqrt{\Delta_\tau \hat{x}^2(i+1) + \Delta_\tau \hat{y}^2(i+1) + \Delta_\tau \hat{\phi}^2(i+1)} \text{ ,(4.14)}$$

$$c(i+1) = \left| v_{ref}(i+1) \right|, \qquad (4.15)$$

where g(i+1) is the Euclidian norm of the change in UGV state caused by the delay τ and c(i+1) is the absolute value of the UGV speed. To optimize the control value $\boldsymbol{u}(i+1)$ it is scaled by a scaling factor K such that the new control value $K\boldsymbol{u}(i+1)$ maximizes the speed while constraining g≤ε which will limit the amount of change in UGV state caused by the delay τ. The optimization problem then becomes:

$$\max_K c(i+1) \text{ subject to } g(i+1) \le \varepsilon \qquad (4.16)$$

With the new control signal being $K\boldsymbol{u}(i+1)$ and using (4.6)-(4.10) we can rewrite (4.14) as:

$$g(i+1) = \sqrt{\frac{2-2\cos(AK\tau)}{A^2 + (AK\tau)^2}}\,. \qquad (4.17)$$

At runtime, after the control signal is generated the values of A and τ_{RTT} are known, and after ε is dynamically calculated using the method described above, the optimization can be carried out using a method such as the bisection algorithm. To increase performance these calculations can be completed offline for known ranges of A, τ_{RTT}, and ε then stored in a three dimensional lookup table. If the curvature, A, is known to be in the range of [0,14] in normal operation and the delay, τ_{RTT}, is known to be in the range of [0,1] then K can be calculated in this range for ε values of [0.1-1]. If ε is allow to go less than 0.1 the UGV will stall and not make progress down the path. The upper limit of ε was chosen to be 1 as it allows the value of K to be 1 for all values of A when τ_{RTT} is close to 0 and for K to be 1 for all values of τ_{RTT} when A is close to 0. The resulting gain tables, some of which are shown in Figure 22, are stored and used as lookup tables during path tracking control.

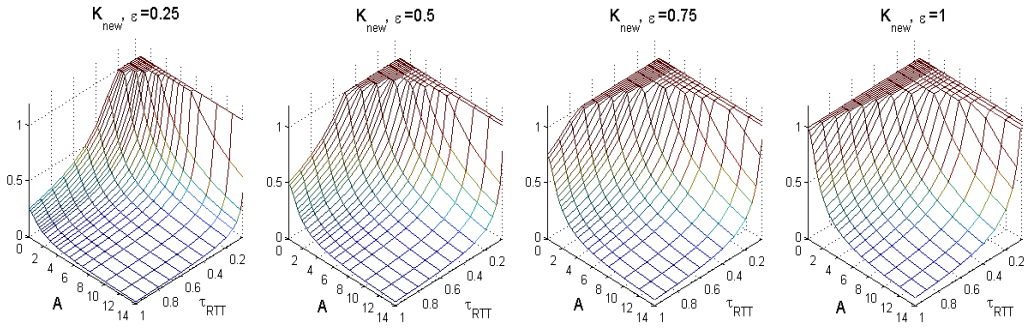


Figure 22: Optimal gain tables for ε values of 0.25, 0.5, 0.75 and 1

The optimal gain surface, shown in Figure 22, will reduce the control signal as delay increases and as the trajectory curvature increases while limiting the deviation of the UGV to ε while operating at the highest speed possible. More discussion on the

gain scheduler can be found in [8].

4.5 Delay Tolerant Behavior Control Based Bandwidth Allocation

When implementing the above method in a multi robot networked control system with bandwidth constraints the performance of individual UGVs can be affected. It is useful to implement a bandwidth allocation method such that the available bandwidth is intelligently allocated to the individual UGVs in a fashion that maximizes the performance of all UGVs in the system.

i) Change in Curvature as a factor for BW Allocation

We propose the allocation of bandwidth according to the change in the curvature of the path and the velocity of the vehicle. At any point in time, the input to the system is linear velocity (v_{ref}) and the angular velocity (ω_{ref}) where the UGV will continue to move in a path of a constant curvature as long as it does not receive the next input command. This suggests that a constant curvature path will be equivalent to a straight path as long as the inputs defined by the linear and angular velocity match the path curvature. The change in the curvature then, is an important factor that should determine the sampling time for the UGV.

ii) Predictive Constrained Gain as an Indicator of Change in the Path Curvature, Network Delay, and Trajectory Curvature

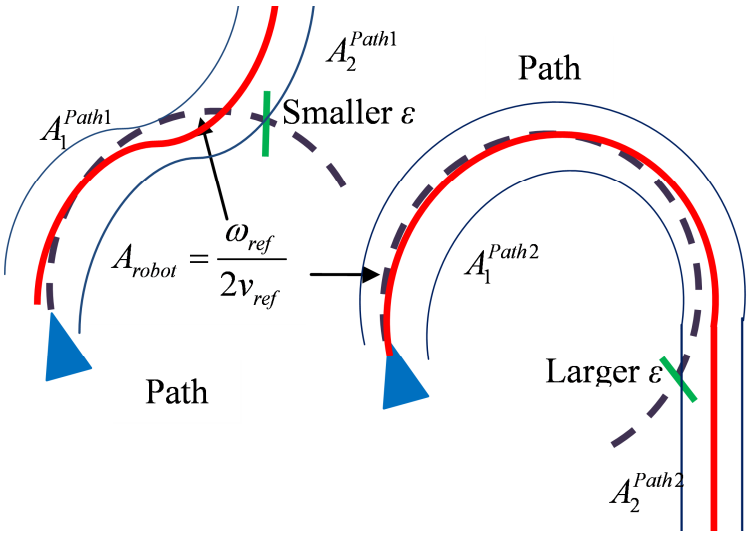


Figure 23: The effect of future change in curvature on ε

In this work we are using predictive gain scheduling for each of the UGVs as described above. The same techniques can be used to predict future curvature and velocity changes in the UGV. The allowable state change, ε, is an indicator of how far the UGV can travel on a given control value such that the it will stay within the constraint of the path. Since each individual control signal is in fact a constant curvature the value of ε will decrease as the curvature of the path ahead changes as seen in Figure 23

Thus, ε indicates the imminence of the change in path curvature. A larger value of ε signifies that the length of the segment of the path which has a curvature that coincides with the curvature of the UGV's trajectory is higher and vice versa. This means that with a higher ε, the urgency for the sampling rate requirements is less. The ε value, however, is not the only factor determining the bandwidth requirement, the UGV trajectory and network delay should also be considered. The

bandwidth requirement of the UGV will increase as the trajectory curvature increases, the delay increases and the ε value decreases. Since the optimal gain, *K*, described in Section III, is an aggregation of these three factors it is an indicator of the bandwidth the UGV requires.

iii) Formulation of Delay Tolerant Behavior Control Approach to Bandwidth Allocation

The stability of the system is defined as

$$\max\left(e(t_i)\right) \le \delta(t_i) \qquad (4.18)$$

where $\delta(t_i)$ is the safety region and $e(t_i)$ is defined as the distance from the UGV coordinates and the nearest path point

Our aim is to find the maximum sampling time that guarantees the stability of the system. This maximum sampling time $\Delta T_{\max}$ can be calculated by solving the optimization problem in (4.19).

$$\max_{\Delta T}\left(e\left(t_i + \Delta T\right)\right) \le \delta(t_i) \qquad (4.19)$$

After putting the system dynamics (4.1) and the error formulation in the optimization problem, we get the following value for $\Delta T_{\max}$.

$$\Delta T_{i,\max} = \frac{B + \sqrt{B^2 - 4DC}}{2D} \qquad (4.20)$$

where,

$$\begin{aligned} B &= 2v(t_i)\left(e_x(t_i) - \Delta p_x(t_i)\right)\cos\phi(t_i) + 2v_i(t)\left(e_y(t_i) - \Delta p_y(t_i)\right)\sin\phi(t_i) \\ C &= e^2(t_i) - \delta^2(t_i) + 2e_x(t_i)\Delta p_y(t_i) + 2e_y(t_i)\Delta p_x(t_i) \\ D &= v^2(t_i) \end{aligned} \qquad (4.21)$$

Thus, the minimum amount of bandwidth required in bits per second (bps) that guarantees stability (4.18) is given by (4.22)

$$BW_{i,\min} = \frac{L \times 8}{\Delta T_{i,\max}} bps \qquad (4.22)$$

where, L is the size of the packet in bytes. The stability of every loop in the system is guaranteed as long as (4.23) is satisfied.

$$\sum_{i=1}^{N} BW_{i,\min} \leq BW_{Total} \qquad (4.23)$$

where N represents the number of control loops. The remaining bandwidth is then calculated using (4.24).

$$BW_r = \min\left(0, \left(BW_{Total} - \sum_{i=1}^{N} BW_{i,\min}\right)\right) \qquad (4.24)$$

Since we are concerned with maximizing the performance of the control loops, the remaining bandwidth is then shared among the control loops (UGVs) by allocating it according to the predicted velocity and the optimal predictive constrained gain K. The amount of bandwidth allocated to each UGV is then calculated using (4.25).

$$BW_i = BW_{i,\min} + \left(\frac{w_1 \times (1 - K_i)}{\sum_j (1 - K_j)} + \frac{w_2 \times \hat{v}_i}{\sum_j (1 - \hat{v}_j)}\right) BW_r \qquad (4.25)$$

This satisfies the bandwidth constraint as long as $w_1 + w_2 \leq 1$ is true.

4.6 Simulation

i) Simulation Scenario

The paths shown in Figure 24 were used for testing the bandwidth allocation algorithm using four UGVs. The simulation was run using Larger Error First (LEF) algorithm [14], Rate Monotonic (RM) Algorithm [12], Behavior Control based (BC-BA) algorithm previously introduced by Ojha and Delay Tolerant Behaviour Control based Algorithm (DTBC-BA) introduced here. For reference purposes, the simulation was also run without any bandwidth allocation. All simulations used predictive constrained gain scheduling.

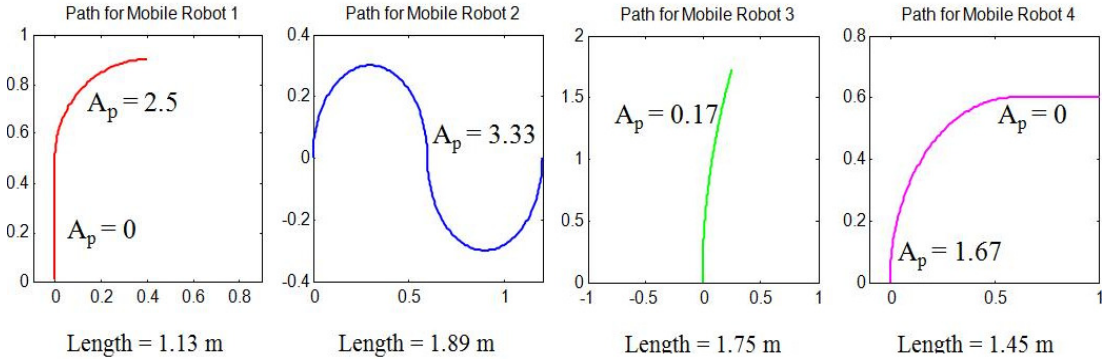


Figure 24: Different Curvature paths used with four different UGVs for testing the performance of each robot.

The simulation was run using 70, 200, 400 and 600ms values for average RTT delay. Two measures were taken to evaluate the dynamic bandwidth allocation algorithms: Trajectory tracking Error (J_1), and tracking time $t_f - t_0$ (J_2). The trajectory tracking error is defined as

$$J_1 = \int_{t_0}^{t_f} (\boldsymbol{a}(t) - \boldsymbol{p}) dt \qquad (4.26)$$

ii) Simulation Results

Figure 25 shows the comparison of the accumulated error at different delays using the different algorithms. As we can see, the error increases as we increase the RTT delay.

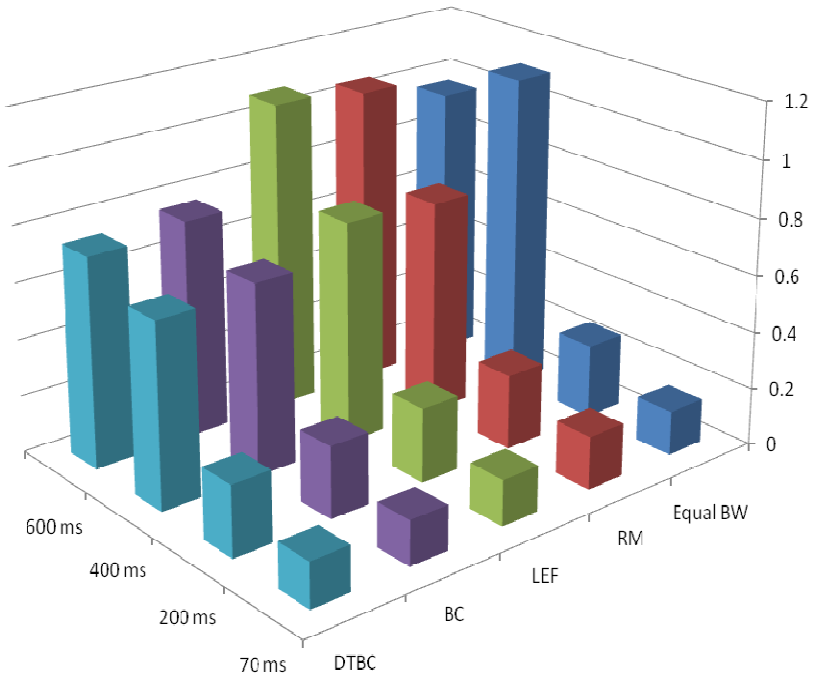


Figure 25: Total accumulated error for different delay conditions using different algorithms

The performance of DTBC-BA compared to the other algorithms is summarized in Table . At lower delay some other algorithms are better but as delay increases the performance of DTBC-BA improves over the other algorithms. This suggests that DTBC-BA is best to use when there is pertinent delay in the network.

Table V: Percentage Improvement in accumulated error when using DTBC as compared to others

Delay	70	200	400	600
Equal BW	-1.30	4.63	41.90	24.78
RM	15.22	5.36	14.15	30.85
LEF	2.50	4.63	16.90	33.11
BC	-1.30	1.20	3.13	5.03

Figure 26 shows the average time it takes to travel a distance of 1m in different delay scenarios when using the five different algorithms. The results again confirm that DTBC-BA performance gets better as the delays become longer.

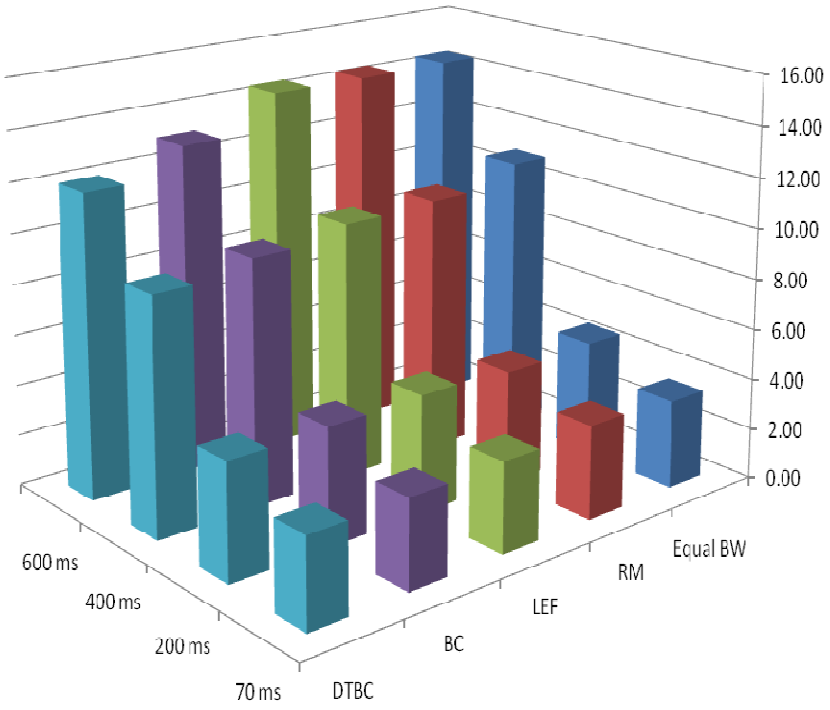


Figure 26: Average time taken to travel a meter in different delay conditions using the different algorithms

Table VI: Percentage Improvement in average time it takes to travel a meter when using DTBC as compared to others

Delay	70	200	400	600
Equal BW	0.19	-0.26	15.21	11.74
RM	5.78	-2.48	9.16	11.57
LEF	1.66	1.94	8.38	11.81
BC	0.11	1.13	3.26	4.75

. A summary of the increase in time performance is given in Table VI.

4.7 Conclusion

The delay and bandwidth allocation issue encountered with the networked path tracking control of multiple UGVs has been compensated for in this work. The predictive constrained path tracking method along with an adaptive behavior control based bandwidth allocation algorithm has been simulated to improve the performance of multiple UGVs sharing the same communication channel in the presence of network delay. Simulated results have shown that by combining predictive control and bandwidth allocation the performance in terms of tracking time has been improved by 50% by using predictive control and the tracking error has been improved by up to 42% using adaptive bandwidth allocation. Simulation results have shown that the proposed method is robust to network delay.

References

[1] C. C. Coulter, "Implementation of the pure pursuitpath tracking algorithm," Robotics Institute, Pittsburgh, PH, Tech-Report January 1992.

[2] K. Yoshizawa, H. Hashimoto, M. Wada, and S. Mori, "Path tracking control of mobile robots using a quadratic curve," in Intelligent Vehicles Symposium, 1996., Proceedings of the 1996 IEEE, 1996, pp. 58-63.

[3] V. Kadakkal and G. Cook, "Use of a preview control scheme with knowledge of future trajectory information for a lane tracking controller on a wheeled mobile robot," in Industrial Electronics, 2008. IECON 2008. 34th Annual Conference of IEEE, 2008, pp. 1692-1697.

[4] S. C. Peters and K. Iagnemma, "Mobile robot path tracking of aggressive maneuvers on sloped terrain," in Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on, 2008, pp. 242-247.

[5] M. Netto, J. M. Blosseville, B. Lusetti, and S. Mammar, "A new robust control system with optimized use of the lane detection data for vehicle full lateral control under strong curvatures," in Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE, 2006, pp. 1382-1387.

[6] S. Glaser, L. Nouveliere, and B. Lusetti, "Speed Limitation Based on an Advanced Curve Warning System," in Intelligent Vehicles Symposium, 2007 IEEE, 2007, pp. 686-691.

[7] R. Vanijjirattikhan, M. Y. Chow, and T. Yodyium, "Feedback preprocessed unmanned ground vehicle network-based controller characterization," in Industrial Electronics Society, 2004. IECON 2004. 30th Annual Conference of IEEE, 2004, pp. 2333-2338 Vol. 3.

[8] R. Vanijjirattikhan, C. Mo-Yuen, P. Szemes, and H. Hashimoto, "Mobile Agent Gain Scheduler Control in Inter-Continental Intelligent Space," in Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on, 2005, pp. 1115-1120.

[9] X. Feng, W. Zhi, and S. Youxian, "Simulation based performance analysis of networked control systems with resource constraints," in Industrial Electronics Society, 2004. IECON 2004. 30th Annual Conference of IEEE, 2004, pp. 2946-2951 Vol. 3.

[10] H. M. Chaskar and U. Madhow, "Fair scheduling with tunable latency: a round-robin approach," Networking, IEEE/ACM Transactions on, vol. 11, pp. 592-601, 2003.

[11] B. A. Forouzan, Data Communications and Networking, 3 ed. Singapore: McGraw Hill, 2004.

[12] J. Lehoczky, L. Sha, and Y. Ding, "The rate monotonic scheduling algorithm: exact characterization and average case behavior," in Real Time Systems Symposium, 1989., Proceedings., 1989, pp. 166-171.

[13] M. S. Brainicky, Philips, S M & Zhang, W. , "Scheduling and Feedback Co-Design for Networked Control Systems," in IEEE/CDC Las Vegas, USA, 2002.

[14] J. Yepez, P. Marti, J. Ayza, and J. M. Fuertes, "LEF closed-loop scheduling policy for real-time control systems," in Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on, 2005, pp. 4 pp.-280.

[15] B. Tao, W. Zhi-ming, and Y. Gen-ke, "Optimal bandwidth scheduling of networked control systems (NCSs) in accordance with jitter," Journal of Zhejiang University - Science A, vol. 6, pp. 535-542, 2005.

[16] G. C. Walsh and Y. Hong, "Scheduling of networked control systems," Control Systems Magazine, IEEE, vol. 21, pp. 57-65, 2001.

[17] L. Zheng and M.-Y. Chow, "Adaptive Multiple Sampling Rate Scheduling of Real-time Networked Supervisory Control System - Part I," in IEEE Industrial Electronics, IECON 2006 - 32nd Annual Conference on, 2006, pp. 4604-4609.

[18] Y. Tipsuwan and M.-Y. Chow, "On the gain scheduling for networked PI controller over IP network," Mechatronics, IEEE/ASME Transactions on, vol. 9, pp. 491-498, 2004.

Chapter 5- Implementation of Delay Tolerant Behavior Control based Adaptive Bandwidth Allocation for Multiple UGVs in a NCS

Unnati Ojha, *Member, IEEE* Bryan R. Klingenberg, *Member, IEEE* Mo-Yuen Chow, *Fellow, IEEE*
Advanced Diagnosis, Automation and Control (ADAC) Laboratory
North Carolina State University
uojha@ncsu.edu, bryanklingenberg@gmail.com, chow@ncsu.edu

5.1 Abstract

In network based path tracking control of systems with multiple unmanned ground vehicles (UGVs), performance can be affected by network constraints including time-varying network delays and bandwidth limitation. Different static as well as dynamic bandwidth management strategies like Larger Error First (LEF), Rate Monotonic (RM), and Behavior Control (BC) Based Allocation etc. have been proposed in the past to allocate bandwidth. However, these existing methods are not robust to network delay which is an important constraint in a Network Control System (NCS). The Delay Tolerant Behavior Control (DTBC) based Adaptive Bandwidth Allocation is one of the bandwidth allocation techniques that is robust to network delays. In this paper, DTBC algorithm has been implemented in an intelligent space environment that uses IEEE 802.15.4 standard wireless protocol for communication between the UGVs and the Supervisory Controller. This method is then compared with other existing bandwidth allocation techniques. The results show that the performance in terms of trajectory tracking error and the time taken to traverse the path decreases sharply in LEF, RM and BC as network delay increases. In contrast, the performance of DTBC is consistent at different network delay conditions.

5.2 Introduction

Many unmanned ground vehicles (UGVs) today communicate wirelessly with remote controllers that make control decisions and shares global data with other controllers. These networked control systems are susceptible to delay which can adversely affect the

performance of the UGVs. This problem becomes especially prominent in a multi-UGV system where multiple UGVs are sharing the same communication channel.

There are many approaches to path tracking control of differential drive UGVs such as pure pursuit [1] and quadratic curve path tracking [2] which track reference points on the path. These path tracking algorithms are effective for non-holonomic locally controlled UGVs. Predictive control has been used in [3] where the predicted trajectory is compared to the future path. Others have used predictive control to compensate for effects of wheel slippage, terrain variations and curvature[4]. In [5] the curvature ahead of the vehicle is detected to limit vehicle speed such that lateral motion will be limited.

The effect of network delay has also been studied. In [6] a feedback preprocessor, similar to a smith predictor, is presented. An elaboration on this approach in [7] adds gain scheduling to prevent a control signal from causing path deviation based on network delay and trajectory curvature.

Network delay is not the only factor that degrades the performance of the NCS. Some other constraints like network bandwidth and CPU processing time may cause the quality of control to degrade to an extent that it may cause instability[8]. Xia, in the same work has also suggested that bandwidth allocation can be an effective method to improve performance of control loops under such conditions. Different types of scheduling algorithms are used to solve the bandwidth allocation problem in networked control systems. Most of these fall under two categories static and dynamic bandwidth allocation. Static methods, like round-robin [9] and priority queuing[10, 11], are not flexible and cannot adapt to the changing environment. To overcome that, adaptive bandwidth allocation techniques have

been introduced. A preemptive rate monotonic (RM) scheduling algorithm is used in[12], where tasks with shorter periods have higher priorities. The sampling period of the NCS is determined by the congestion levels in the network. Large Error First (LEF) technique, used in[13], uses feedback information from the control loop to determine the bandwidth to be assigned to each individual component. In this paper, the bandwidth allocation problem is confronted from the controlled system's performance perspective.

A predictive gain scheduler presented by Klingenberg, Ojha and Chow in [14] where the future position, along with the future path, is used to determine how far the UGV will accurately track the path given a current state and control signal. In another work by Ojha, Klingenberg and Chow[15], a Behavioral Control (BC) approach is presented that uses the current curvature and velocity of the UGVs for bandwidth allocation. In [16] Klingenberg, Ojha and Chow have further improvised the BC approach into a Delay Tolerant Behavioral Control (DTBC) approach for bandwidth allocation. This method uses predictive gain instead of curvature to dynamically allocate bandwidth for each UGV. Larger predictive gain values indicate that each UGV can travel farther on the current control value requiring less bandwidth and vice versa.

The remaining sections are organized as follows: Section II provides the description of the UGV, controller and the DTBC bandwidth allocation method. Section III describes the test bed and the test conditions that were used to implement the DTBC method. The performance of DTBC is compared with LEF, RM and BC at different delay scenarios in section V. Section VI concludes the paper.

5.3 System Description

i) UGV Dynamics

The distributed network control system in this research is the path tracking control of multiple UGVs from a central supervisory controller over a network. Each UGV is a differential drive UGV with two driving wheels and one caster wheel. The dynamics of the UGV can be described using the kinematic model in (4.1):

$$\begin{bmatrix} v_{ref} \\ \omega_{ref} \end{bmatrix} = \begin{bmatrix} \frac{\rho}{2} & \frac{\rho}{2} \\ \frac{\rho}{W} & -\frac{\rho}{W} \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}, \tag{5.1}$$

where $\boldsymbol{v}_{ref}$ is the linear velocity of the UGV, and $\boldsymbol{\omega}_{ref}$ is the angular velocity of the UGV, ρ is the radius of the drive wheels and W is the distance between the drive wheels. ω_r and ω_l represent the angular velocities of the right and left drive wheel respectively.

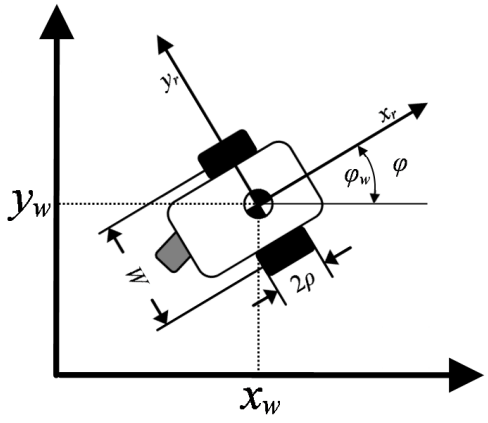


Figure 27: Differential drive mobile robot reference frame and parameters

The ω_r and ω_l variables above represent the angular velocities of the right and left drive wheel respectively. The UGV steers by driving the wheels at different

speeds determined by solving (4.1) where $\boldsymbol{v}_{ref}$ and $\boldsymbol{\omega}_{ref}$ are given as an input reference command $\boldsymbol{u}=[\boldsymbol{v}_{ref}\ \boldsymbol{\omega}_{ref}]^{\mathrm{T}}$

$$\begin{bmatrix} \omega_{R,ref} \\ \omega_{L,ref} \end{bmatrix} = \begin{bmatrix} \frac{1}{\rho} & \frac{W}{2\rho} \\ \frac{1}{\rho} & -\frac{W}{2\rho} \end{bmatrix} \begin{bmatrix} v_{ref} \\ \omega_{ref} \end{bmatrix}. \qquad (5.2)$$

Using (5.2) the input reference command can be used to set reference speeds for the individual wheels. A PI controller, defined by(5.3), is used to achieve the reference speed on each wheel,

$$\mathbf{u}(t) = K_P \mathbf{e}(t) + K_I \int_0^t \mathbf{e}(\xi) d\xi\,, \qquad (5.3)$$

where KP is the proportional gain, KI is the integral gain, and $e(t)$ is the difference between the actual wheel speeds and the reference wheel speeds as shown in (5.4)

$$\mathbf{e}(t) = \begin{bmatrix} \omega_R - \omega_{R,ref} \\ \omega_L - \omega_{L,ref} \end{bmatrix}. \qquad (5.4)$$

The two control parameters in (5.3) are used to control the three UGV states defined by (4.2)

$$\mathbf{a}(t_i) = \begin{bmatrix} x(t_i) & y(t_i) & \phi(t_i) \end{bmatrix}, \qquad (5.5)$$

where $x(t_i)$ and $y(t_i)$ are the UGV coordinates in the world frame and $\varphi(t_i)$ is the UGV heading expressed as the angle between the UGV and the positive x axis of the world frame at time t_i.

ii) Path Tracking Controller

Quadratic curve (QC) path tracking controller defined in [2] along with Feedback Pre-processor (FP) and Predictive Control Gain Scheduling Middleware

(PCGSM) [16] is used for controlling each UGV. The controller calculates a path, $\boldsymbol{p}$, defined by a set of consecutive waypoints on the ground for each UGV to follow as $\boldsymbol{p} = [\boldsymbol{p_x}\ \ \boldsymbol{p_y}]$ where $\boldsymbol{p_x}$ and $\boldsymbol{p_y}$ are sets of world frame *x*- and *y*- coordinates respectively. QC path tracking algorithm will determine a look-ahead distance based on the previous UGV control. A reference point ahead of the UGV is constrained to move along $\boldsymbol{p}$. The location of the reference point is determined by finding the nearest point on the path to the UGV and looking ahead in the path a variable distance defined using methods in [2].

To track the reference point, the controller finds a quadratic curve that passes through the reference point and has its vertex located at the UGV. Since the sampling period is sufficiently small, reference velocity $\boldsymbol{v}_{ref}$ and turn rate $\boldsymbol{\omega}_{ref}$ can then be calculated by fitting a circle in the quadratic curve [2] as:

$$A = \frac{e_y}{e_x^2}, \tag{5.6}$$

$$v_{ref} = \mathrm{sgn}(e_x)\frac{\alpha}{1+|A|}, \tag{5.7}$$

$$\omega_{ref} = 2Av_{ref}, \tag{5.8}$$

where A is the quadratic curve coefficient, α is the maximum velocity. Using A to determine the velocity causes the UGV to slow down while turning to prevent error. Since the reference command is a circular trajectory defined by (4.4) and (4.5), the controller needs to frequently generate new quadratic curves and subsequent circular arcs for the UGV. The UGV will follow the reference circular trajectory for a short

period of time such that when all of the circular arcs are aggregated together, they approximate the quadratic curve which in turn approximates the desired path.

iii) Delay Tolerant Behavior Control (DTBC) Based Bandwidth Allocation

The DTBC algorithm allocates the bandwidth according to the change in the curvature of the UGV trajectory, the amount of delay in the network and the velocity of the UGV.

The predictive gain technique introduced in [14] is used in [16] to predict future curvature and velocity changes in the UGV. The allowable state change, ε, is an indicator of how far the UGV can travel on a given control value before it violates the path constraint. Since each individual control signal is in fact a constant curvature, the value of ε will decrease as the curvature of the path ahead of the UGV changes.

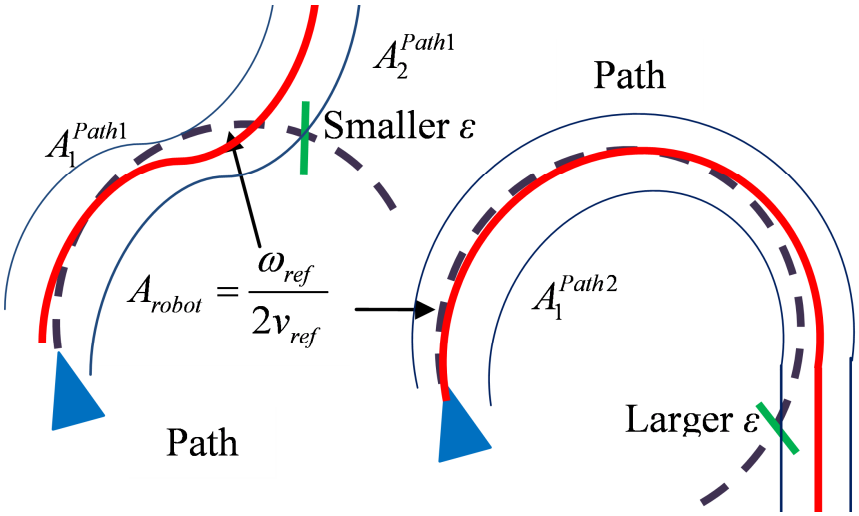


Figure. 28. The effect of future change in curvature on ε.

As we can see in Figure 23, with the same linear velocity ($\boldsymbol{v}_{ref}$) and the angular velocity ($\boldsymbol{\omega}_{ref}$), the UGV can travel farther on Path 2 compared to Path 1 without

violating the safety constraint because Path 2 has a longer section with constant curvature. Thus, ε is a good indicator of future change in path curvature. A larger ε signifies that the length of the segment of the path with a curvature coinciding with the curvature of the UGV's trajectory is higher, and vice versa. With a large ε, the urgency for the sampling rate requirement is lesser. The ε value, however, is not the only factor determining the bandwidth requirement. The bandwidth requirement of the UGV will increase as the trajectory curvature increases, as the delay increases, and as the ε value decreases, respectively. Since the optimal gain, K, described in [14], is an aggregation of these three factors, it is a good indicator of the bandwidth the UGV requires.

In DTBC, a stability criterion is first defined as:

$$\max\left(e(t_i)\right) \le \delta(t_i), \qquad (5.9)$$

where $\delta(t_i)$ is the safety region around the path and $e(t_i)$ is defined as the distance from the UGV coordinates to the nearest path point. The maximum sampling time that guarantees the stability of the system, ΔT_{max}, is then calculated by solving the algebraic equation:

$$e\left(t_i + \Delta T_{\max}\right) = \delta(t_i). \qquad (5.10)$$

Thus, the minimum amount of bandwidth required in bits per second (bps) that guarantees stability (4.18) is:

$$BW_{i,\min} = \frac{L \times 8}{\Delta T_{i,\max}} bps, \qquad (5.11)$$

where L is the size of the packet in bytes. The stability of every loop in the system is guaranteed as long as (4.23) is satisfied:

$$\sum_{i=1}^{N} BW_{i,\min} \le BW_{Total}, \qquad (5.12)$$

where N is the number of control loops. The remaining bandwidth is then shared among the control loops (UGVs) by allocating it according to the predicted velocity and the optimal predictive gain K. The amount of bandwidth allocated to each UGV is then calculated as:

$$BW_i = BW_{i,\min} + \frac{1}{2}\left(\frac{(1-K_i)}{\sum_j (1-K_j)} + \frac{\hat{v}_i}{\sum_j \hat{v}_j} \right) BW_r, \qquad (5.13)$$

which satisfies the bandwidth constraint (4.23).

5.4 Test-bed Description and Test Conditions

i) Intelligent Environment

A network based integrated navigation system has different modules combined together to guide a UGV from one point to another in the space of interest where the navigational intelligence comes from a remote controller. The different modules in the space of interest can be combined together using the intelligent space concept. The intelligent space (iSpace) concept has been used to effectively implement distributed sensors, distributed actuators, distributed processors, and information technology over physically connected space and spaces connected via communication networks [17].

The intelligent space implemented in this project has a network of distributed sensors (cameras and encoders) and multiple actuators (UGVs). iSpace aggregates the entire space status from each agent's sensory information and responds intelligently to the system goals. The mobile agent in this implementation is a UGV that is controlled by iSpace to move between locations tracking a path to avoid obstacles.

ii) The TriSPOT Mobile Robot

The differential drive mobile robot used in this platform has been constructed out of modular off the shelf LEGO Mindstorms NXT TriBot which provides an easy to use set of pieces and convenient motors with encoders. The NXT controller, however, was replaced with a Small Programmable Object Technology (SPOT) controller from Sun Microsystems. The Sun SPOT controller provides general purpose I/O pins and an 802.15.4 radio with antenna. To interface the Sun SPOT and the NXT motors an H-Bridge circuit was built to amplify the PWM output from the Sun SPOT. To process the quadrature data from the two encoders a PIC was used to offload this task from the Sun SPOT. To communicate the data back to the Sun SPOT a UART was used. The block diagram showing these connections can be seen in Figure 29.

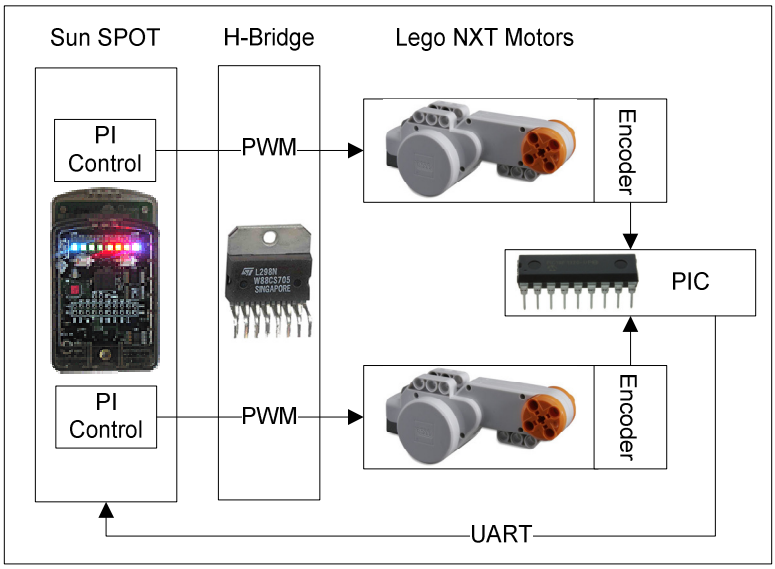


Figure 29: TriSPOT hardware schematic

Since the UGV described above is a combination of the LEGO TriBot and Sun SPOT it has been named a TriSPOT. The TriSPOT has a wheel radius, ρ, of 0.0286m and distance between wheels, W, of 0.095m. With the gearing of the NXT motors the maximum speed for this UGV is approximately 0.4m/s.

iii) Supervisory Controller and Communications

The supervisory controller is used to combine the distributed sensor information and make control decisions to accomplish the system goals. The supervisory controller is implemented using a Sun SPOT base station connected to a host PC over USB. The base station is used to communicate with the Sun SPOT on the TriSPOT. The host PC is also connected to the internet for access to the image processing server. The main control and feedback path is carried over the 802.15.4 wireless radio stream between the TriSPOT and the base station with supplementary

data coming from the image processing server. The extent of the different communication paths, types of communications, and programming languages used can be seen below in Figure 30.

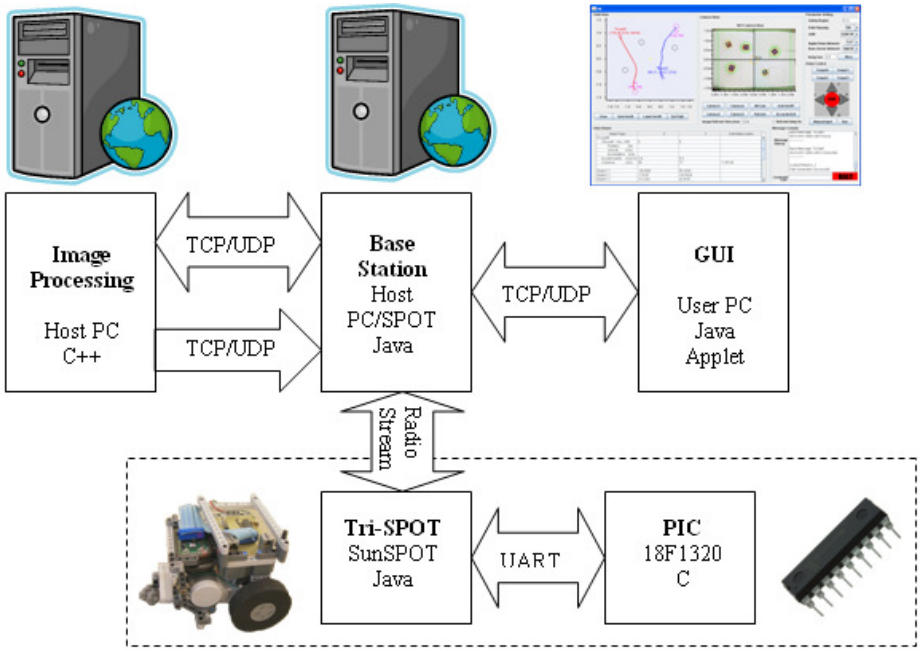


Figure 30: System architecture showing communication paths

In addition to implementing the data fusion and connecting distributed sensors, the supervisory controller also makes control decisions. The quadratic curve path tracking algorithm, described below, is implemented on the supervisory controller. Predictive delay compensation is also implemented on the supervisory controller.

To provide a user interface for the system a Graphical User Interface (GUI) has been developed. The GUI has been written as a java applet to allow it to be embedded into any web so that with proper access and security clearance the system

can be controlled from any location with internet access. The GUI communicates directly to the base station (supervisory controller) Through the GUI the user can control the connections to the UGV and the image processing server. The user can issue manual commands to each of these and display data acquired from both as well. Using the stitched camera images and displayed data from the image processing server the user is able to select the UGV and destination. When the user commands the UGV to follow the path the progress and status of the UGV is updated on the GUI while the UGV is tracking the path. Finally, the GUI also allows for manual control and tuning of the UGVs through a command interface.

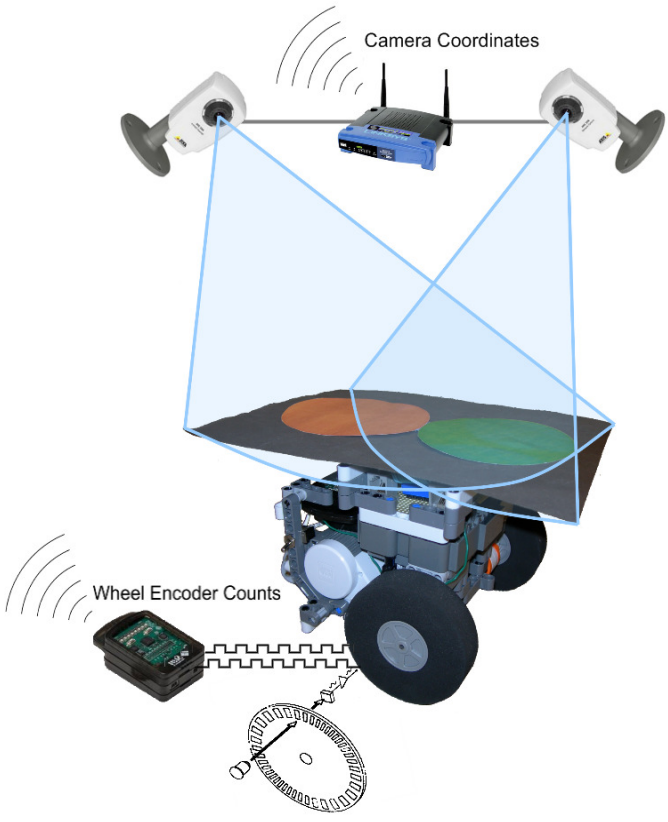


Figure 31: Distributed Sensors in iSpace

iv) Test Paths

Four Test paths as shown in Figure 32 are used for the four UGVs which represent different curvature scenarios in order to test the implemented method. The UGVs will be commanded to follow test paths using No Bandwidth Allocation, LEF, RM, BC and DTBC with varying network delays as characterized below. We will compare the effects of roundtrip time delays of 60ms delay, 200ms, 400ms, and 600ms. The same paths were used for simulation results thus another reason for using these paths is also to compare the simulation and the experimental results.

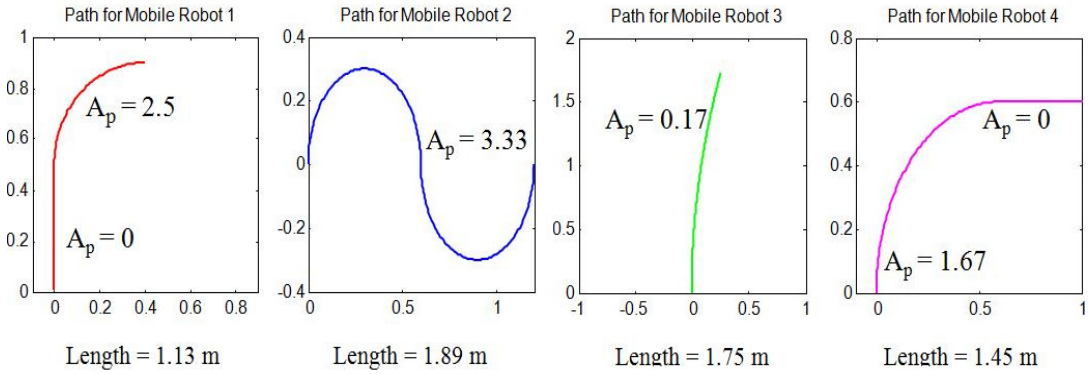


Figure 32: Test Paths

v) Network Delay

The UGVs in consideration are controlled over a wireless network that causes the control signals and feedback signals to be delayed. This delay can be represented as the time that a signal takes to be transmitted from the base station (controller) to the TriSPOTs (UGV), τ_{CR}, and the time that a signal takes to be transmitted from the UGV to the controller, τ_{RC}. The sum of these delay times is the round trip delay, $\tau_{RTT} = \tau_{CR} + \tau_{RC}$. The communication between the base station and TriSPOT is over and

IEEE 802.15.4 network. The delays associated with this type of network have been characterized in [18] and [19]. In this work the radio is operating at 2.4 GHz and 250 Kbps and since 802.15.4 is a beacon enabled network, this results in a minimum beacon interval of 15ms. The network delay will then be in multiples of the 15ms beacon interval.

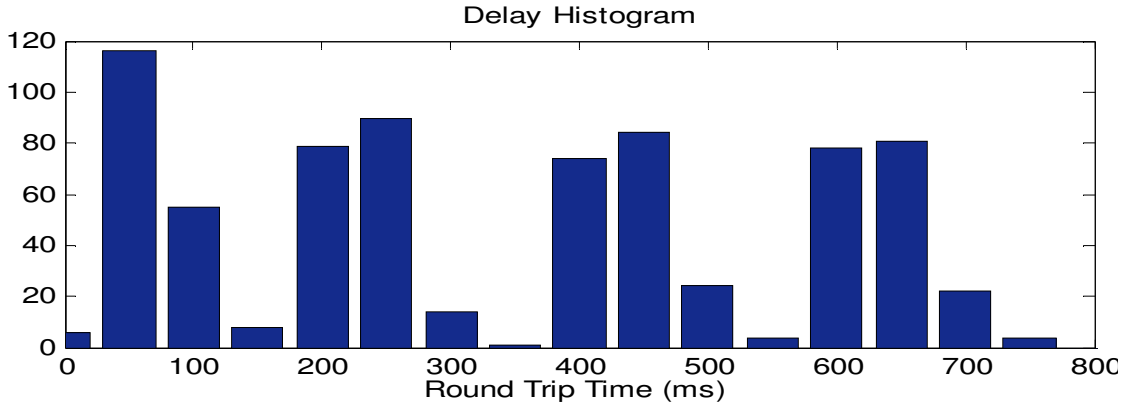


Figure 33: Network delay histogram for delay generator RTT settings of 60, 200, 400, and 600 ms

In order to facilitate testing, a software based delay generator was also implemented. This delay generator acts on control signals and feedback to replicate the effects of network delay. To characterize the delay in this system the TriSPOT was pinged repeatedly with different delay settings for the delay generator. With the delay generator turned on to 200ms it will add 200ms of delay to the each direction of data. The results shown in Figure 33 indicate that when the delay generator is turned on at 60ms, 200ms, 400ms and 600ms, the mean delay were 73ms, 237ms, 445ms and 647ms respectively.

vi) Performance Metrics

To measure the performance of the UGVs and to allow for comparison between different bandwidth allocation methods, the following performance metrics have been implemented. First, the accumulated error is calculated as the area between the desired path and the actual path of the UGVs and is calculated as shown in(5.14).

$$J_1 = \sum_{i=0}^{N} \left(\int_{t_{i,0}}^{t_{i,f}} \left(\boldsymbol{y}_i(t) - \boldsymbol{p}_i \right)^2 dt \right), \qquad (5.14)$$

where N is the number of UGVs. The second performance metric, shown in(5.15), is simply the average time taken to travel 1m for the UGVs.

$$J_2 = \frac{\sum_{i=0}^{N} \left(\frac{t_{i,f} - t_{i,0}}{\text{length of path i}} \right)}{N}. \qquad (5.15)$$

5.5 Results from Implementation

Figure 34 shows the amount of bandwidth allocated to each TriSPOT at the roundtrip time delay of 200ms. RM and Static bandwidth allocation techniques allocate constant amount of bandwidth throughout the run time and thus are not included in the graph. Static bandwidth allocates 25% bandwidth to each TriSPOT and RM allocates 26%, 30%, 14% and 30% bandwidth to each TriSPOT respectively. In LEF, TriSPOT 4 marked by the region in magenta gets around 40% of the bandwidth during its runtime thus we can observe in the error plot that TriSPOT 4 has relatively lesser error when LEF is used. At around time t=2s, LEF allocates almost all of the bandwidth to TriSPOT which caused TriSPOT 1 to move away from the path thus increasing the amount of error even though after time t=3s, TriSPOT

1 finally gets around 30% of the bandwidth. Unlike BC and DTBC, TriSPOT 1 finishes after TriSPOTs 3 and 4 in LEF. In BC, the amount of bandwidth allocated is almost the same for all the TriSPOTs. Note that TriSPOT 2 gets the maximum amount of bandwidth all along which could be because of the sharp curvature of path 2.

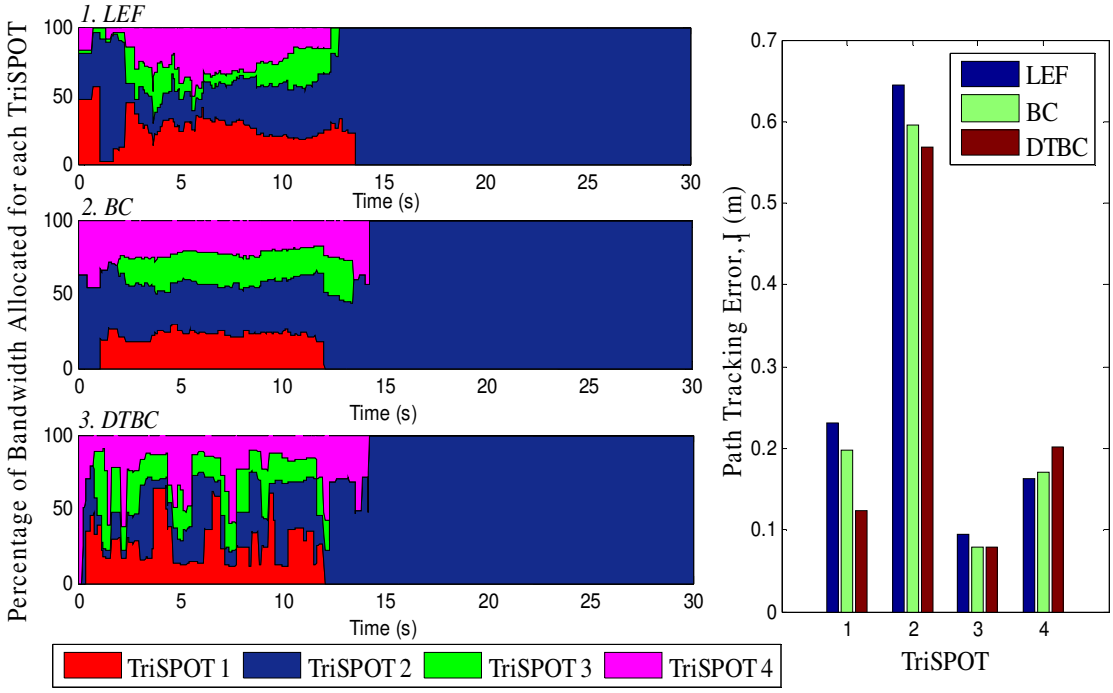


Figure 34: Amount of bandwidth allocated for each TriSPOT using LEF, BC and DTBC and the corresponding accumulated error

In case of DTBC, the amount of allocated bandwidth for each TriSPOT changes rapidly, which could be attributed to the change in the epsilon value. The epsilon value reflects the amount change in the TriSPOT trajectory curvature. As the controller detects a change in TriSPOT trajectory, it allocates more bandwidth to that TriSPOT until the curvature of the TriSPOT trajectory becomes stable. The spikes represent the amount of

bandwidth allocated during that transient time. Though there are spikes in the amount allocated bandwidth for each TriSPOT, in average, TriSPOT 2 gets the most bandwidth which is again because of the sharp curvature of path 2. We can clearly see that TriSPOT 4 gets a relatively lesser amount of bandwidth thus the accumulated error is greater in this case as compared to the other two.

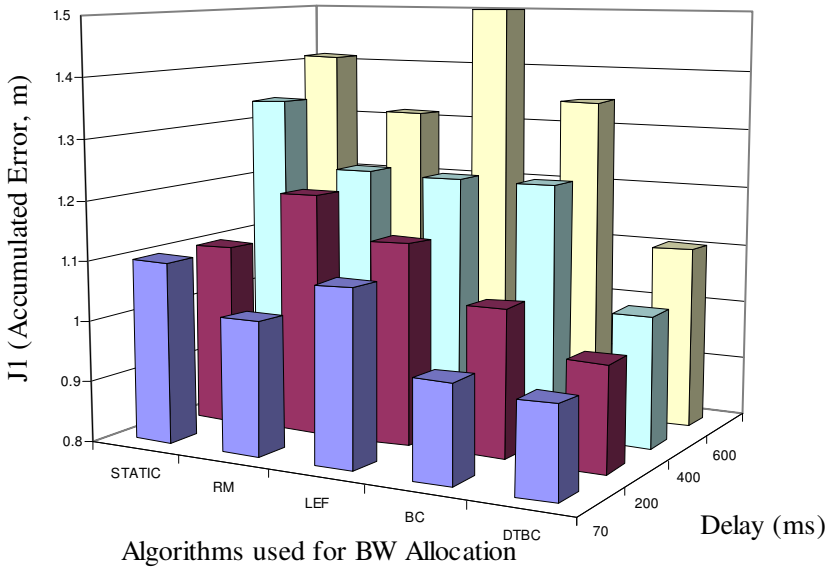


Figure 35: Accumulated error for different delay conditions using different algorithms

Figure 35 shows the accumulated error J_1 at different delays using the algorithms. The error increases with delay however in DTBC the error increases at a lower rate compared to other algorithms. Also note that BC and DTBC perform almost similar at lower delay values but the performance of BC soon degrades with a greater rate as the delay increases. Table VII shows the amount of improvement in the performance of DTBC as compared to the other algorithms at different delays. This value was computed as:

$$\%\ improvement = \frac{J_{1,\text{algorithm i}} - J_{1,DTBC}}{J_{1,\text{algorithm i}}} \times 100 \quad (5.16)$$

Table VII: Percentage Improvement in accumulated error J_1 when using DTBC as compared to others.

Delay	STATIC	RM	LEF	BC
70	13.92	7.06	12.94	1.36
200	11.59	18.97	14.16	6.70
400	23.97	16.85	16.68	16.54
600	21.79	16.00	43.33	18.02

Clearly, Table VII suggests that DTBC performs much better than other algorithms in scenarios with or without delay. DTBC is more robust to network delays.

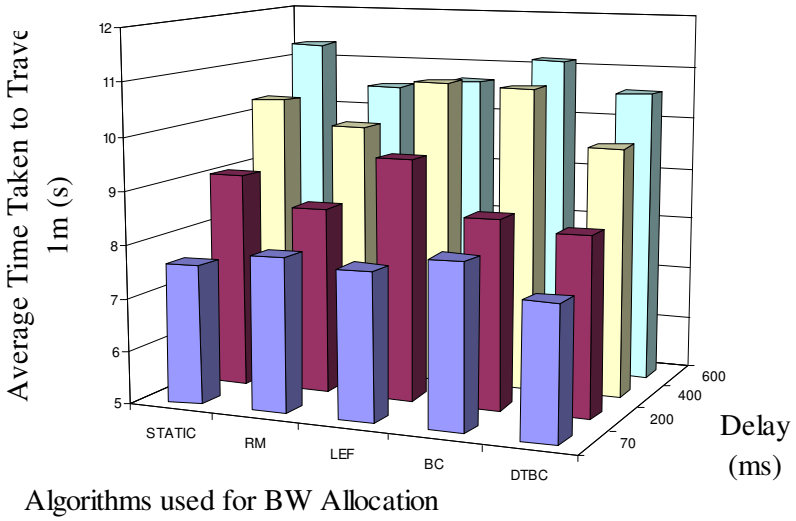


Figure 36: Average time taken to travel a meter in different delay conditions using the different algorithms

As can be seen in Figure 36, DTBCs performance is comparable to other algorithms in terms of average time taken to travel 1m. In fact, though by a very small amount, J_2 is also

better in case of DTBC. This is clarified further by Table VIII. The calculation of % improvement is similar to (5.16) except that J_2 is used instead of J_1.

Table VIII: Percentage Improvement in average Time when using DTBC as compared to others

Delay	STATIC	RM	LEF	BC
70	1.81	5.15	3.72	7.34
200	7.43	1.75	12.21	2.11
400	5.97	1.61	9.99	9.55
600	6.56	-0.56	1.02	5.14

5.6 Conclusion

This paper has presented the path tracking performance for four bandwidth allocation methods in terms of tracking error and average tracking. The performance of DTBC is compared with the existing methods Static, RM, LEF and BC. The experimental results have shown that the performance improvement when using DTBC in terms of accumulated error goes up to 23% compared to Static, 18% compared to RM, 43% compared to LEF and 18% compared to BC. In terms of average time taken to travel 1m, DTBC is comparable to other algorithms. The UGVs could travel up to 7.5%, 5%, 12% and 10% faster compared to Static, RM, LEF and BC respectively when using DTBC approach to bandwidth allocation. The results show DTBC approach is robust to network delays.

Acknowledgment

This research was supported under NSF-ECS-0823952 “Impaired Driver Electronic Assistant (IDEA)” project and DOT Massive Sensor Based Congestion Management System for Transportation System projects.

References

[1] C. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," Robotics Institute, Pittsburgh, PH, Tech-Report January 1992.

[2] K. Yoshizawa, H. Hashimoto, M. Wada, and S. Mori, "Path tracking control of mobile robots using a quadratic curve," in *Intelligent Vehicles Symposium, 1996., Proceedings of the 1996 IEEE*, 1996, pp. 58-63.

[3] V. Kadakkal and G. Cook, "Use of a preview control scheme with knowledge of future trajectory information for a lane tracking controller on a wheeled mobile robot," in *Industrial Electronics, 2008. IECON 2008. 34th Annual Conference of IEEE*, 2008, pp. 1692-1697.

[4] S. C. Peters and K. Iagnemma, "Mobile robot path tracking of aggressive maneuvers on sloped terrain," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 2008, pp. 242-247.

[5] M. Netto, J. M. Blosseville, B. Lusetti, and S. Mammar, "A new robust control system with optimized use of the lane detection data for vehicle full lateral control under strong curvatures," in *Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE*, 2006, pp. 1382-1387.

[6] S. Glaser, L. Nouveliere, and B. Lusetti, "Speed Limitation Based on an Advanced Curve Warning System," in *Intelligent Vehicles Symposium, 2007 IEEE*, 2007, pp. 686-691.

[7] R. Vanijjirattikhan, M. Y. Chow, and T. Yodyium, "Feedback preprocessed unmanned ground vehicle network-based controller characterization," in *Industrial Electronics Society, 2004. IECON 2004. 30th Annual Conference of IEEE*, 2004, pp. 2333-2338 Vol. 3.

[8] X. Feng, W. Zhi, and S. Youxian, "Simulation based performance analysis of networked control systems with resource constraints," in *Industrial Electronics Society, 2004. IECON 2004. 30th Annual Conference of IEEE*, 2004, pp. 2946-2951 Vol. 3.

[9] H. M. Chaskar and U. Madhow, "Fair scheduling with tunable latency: a round-robin approach," *Networking, IEEE/ACM Transactions on,* vol. 11, pp. 592-601, 2003.

[10] B. A. Forouzan, *Data Communications and Networking*, 3 ed. Singapore: McGraw Hill, 2004.

[11] J. Lehoczky, L. Sha, and Y. Ding, "The rate monotonic scheduling algorithm: exact characterization and average case behavior," in *Real Time Systems Symposium, 1989., Proceedings.*, 1989, pp. 166-171.

[12] M. S. Brainicky, Philips, S M & Zhang, W. , "Scheduling and Feedback Co-Design for Networked Control Systems," in *IEEE/CDC* Las Vegas, USA, 2002.

[13] J. Yepez, P. Marti, J. Ayza, and J. M. Fuertes, "LEF closed-loop scheduling policy for real-time control systems," in *Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on*, 2005, pp. 4 pp.-280.

[14] B. R. Klingenberg, U. Ojha, and M.-Y. Chow, "Predictive Constrained Gain Scheduling for UGV Path Tracking in a Networked Control System," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009.

[15] U. Ojha, B. R. Klingenberg, and M.-Y. Chow, "Dynamic Bandwidth Allocation based on Curvature and Velocity in a Fleet of Unmanned Ground Vehicles," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009.

[16] B. R. Klingenberg, U. Ojha, and M.-Y. Chow, "Predictive Control of Multiple UGVs in a NCS with Adaptive Bandwidth Allocation," in *35th Annual Confrence of the IEEE Industrial Electronics Society (IECON '09)* Porto, portugal, 2009.

[17] Y. Tipsuwan and M.-Y. Chow, "On the gain scheduling for networked PI controller over IP network," *Mechatronics, IEEE/ASME Transactions on,* vol. 9, pp. 491-498, 2004.

[18] L. Jin-Shyan, S. Yu-Wei, and S. Chung-Chou, "A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi," in *Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE*, 2007, pp. 46-51.

[19] M. Petrova, J. Riihijarvi, P. Mahonen, and S. Labella, "Performance study of IEEE 802.15.4 using measurements and simulations," in *Wireless Communications and Networking Conference, 2006. WCNC 2006. IEEE*, 2006, pp. 487-492.