Remote Low Frequency State Feedback Kinematic Motion Control for Mobile Robot Trajectory Tracking

Daniel Montrallo Flickinger and Mark A. Minor

***Abstract*— Teleoperated robots generally receive high level commands from a remote system, while accomplishing motion control through conventional means. We present a teleoperated system that removes the entire motion control structure from the robot, in order to preserve the availability of crucial on-board resources. The operation of state feedback control is performed by a system remote from the robot. We have designed a computerized motion planning and control system for Mobile Emulab, and in this article, discuss the implementation of trajectory tracking control. A component of the Emulab network testbed, Mobile Emulab is used for wireless network experiments requiring mobility; and is publicly available to remote researchers via the Internet.**

Medium scale wheeled mobile robot couriers are used to move wireless antennas within a semi-controlled environment. Experimenters use a web-based GUI to specify desired paths and configurations for multiple robots.

State feedback is provided by an overhead camera based visual localization system. Kinematic control is used to generate velocity commands, which are sent to robots over a computer network. Data availability is restricted to a low sampling frequency. There is significant noise, loss, and phase lag present in the robot localization data, which our research overcomes to provide an autonomous trajectory tracking mobile robot control system.

***Index Terms*— Mobile robots, Motion control, Telerobotics**

I. Introduction

This research addresses the engineering challenges encountered while designing and implementing a system providing teleoperation and remote feedback control of mobile robots over a network. The motion control system presented as part of this research is added as a component to Mobile Emulab [12], which is part of the Emulab network testbed [22].

An overview of the Mobile Emulab system architecture is shown in Figure 1. The major components of Mobile Emulab are displayed, with their interactions denoted. The motion controller passes desired wheel velocities to the robots. The cameras pass image data to the localization system, which is used to determine the positions of all robots in the testbed environment. Robots are controlled over a wireless network and localized by a computer vision system which utilizes an overhead grid of video cameras.

The motion controller presented in this paper is motivated by the need for smooth, precise motion control of robots

Largely sponsored by NSF grants CNS-0335296 and EIA-0321350

D. M. Flickinger is currently with the Department of Aerospace and Mechanical Engineering, University of Notre Dame, Notre Dame, IN 46556, USA Formerly with the Department of Mechanical Engineering, University of Utah flikx@flux.utah.edu

M. Minor is with the Department of Mechanical Engineering, University of Utah, Salt Lake City, UT 84112, USA minor@mech.utah.edu

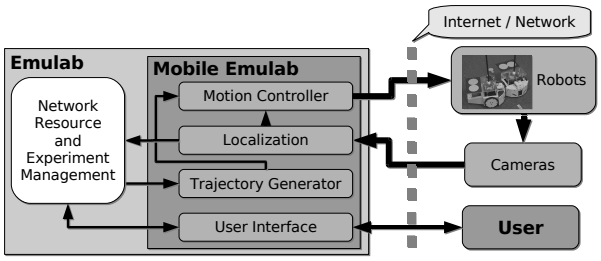

Fig. 1. Mobile Emulab system architecture overview.

in the Mobile Emulab wireless network testbed. Mobile Emulab is designed to provide wireless network researchers a useful and convenient platform for conducting mobile and wireless studies in real radio environments. This platform frees researchers from the inherent inaccuracies of wireless simulation, which is often used to evaluate new mobile wireless network protocols and applications. Furthermore, it reduces the barrier to real mobile wireless research by providing an autonomous motion control system.

There exists demand in the wireless networking research community to conduct experiments on real world hardware, as opposed to virtual simulation of wireless signal properties. Moving wireless precisely and frequently is a tedious task for humans, but is well suited for mobile robots. Emulab provides features useful for the creation and administration of networking experiments. Existing users of the Emulab testbed are located globally, and must be able to place wireless networking equipment within the testbed environment remotely over the Internet.

Commercially available Acroname Garcia robots [7], are used as couriers of wireless network interfaces, antennas, sensors, and a small single-board computer. Limited computational resources on the robots must be reserved for user applications needed to perform and administer experiments. This constraint requires that we must implement motion control on a system remote from the actual robot hardware. Localization data cannot be provided at a rate faster than 30 Hz, constrained by camera frame rate.

The organization of this article is as follows: Background about the field and how it relates to this research is given in Section II. An overview of the chosen motion controller is in Section III. The methods used to implement the controller into a remote robot control system are discussed in Section IV, and the simulation and experimental results of this work are given in Section V. Concluding remarks are in Section VI.

II. Background

Motion control of mobile robots initially centered on kinematic techniques. The main focus during the early 1990s was on providing ideal velocity commands that could provide posture regulation or trajectory tracking in consideration of nonholonomic constraints. Approaches were based upon time varying and discontinuous control laws that satisfied Brockett's theorem [15]. Polar coordinates were then introduced in order to provide smooth time invariant kinematic controllers that could provide posture regulation and possibly trajectory tracking [1], [11]. Given the difficulty of reproducing these velocities on actual robots, the focus shifted in the late 1990s to backstepping based controllers that considered kinematic control [13] in conjunction with dynamic controllers to provide wheel torque commands [5]. A variety of robust and adaptive controllers were then examined during the early 2000s. Subsequent research has focused on providing smooth time kinematic controllers capable of satisfying physical constraints [14] in conjunction with robust dynamic controllers capable of rejecting disturbances [24], [25]. The Garcia robots used in this research are based upon embedded velocity servo loops, however, and are not amenable to these more advanced dynamic controllers that typically require higher sampling rates and torque commands. While any of a variety of kinematic motion controllers could have been used in this research, the path-manifold kinematic controller presented in [24] and described in further detail in [14] was implemented since it considers physical constraints and provides velocity commands suitable for our robots.

Mobile robot testbeds [4], [10], [12] are used to evaluate high level coordination and motion planning, or to run specific experiments, such as wireless network evaluation. A common element in these systems is the design of a centralized control system. In most examples, coordination and high level motion commands are issued to autonomous robots in the testbed. These robots may self localize, and possess local motion controllers. In our research, we have developed a system where the capabilities of the individual robots are limited. No local motion control is present, and all localization is handled centrally. In our system, robots send velocity commands received over a network directly to wheel level controllers, minimizing the amount of computational resources required on each robot. This maximizes the amount of resources available for experimentation.

Teleoperation over the Internet [17], pioneered in [9], involves the operation of robots by remote users issuing high level commands. Web-based interfaces allow users to coordinate robot motions [20]. For example, mobile robots may be remotely controlled through haptic interfaces [16]. We extend this model, and substitute the human-controlled haptic interface with a computer system running a motion controller to execute a predetermined trajectory. Teleoperation of mobile robots using commercially available wireless networking hardware is possible [19], [21]. Our system advances this to use non-dedicated networks, with unpredictable characteristics. We also solve control stability problems in this environment. We extend teleoperation to separate all motion control functions from the robot. With our system, not only can a user be remote, but almost all of the software driving a robot can be at a completely different location as both the user and the robot.

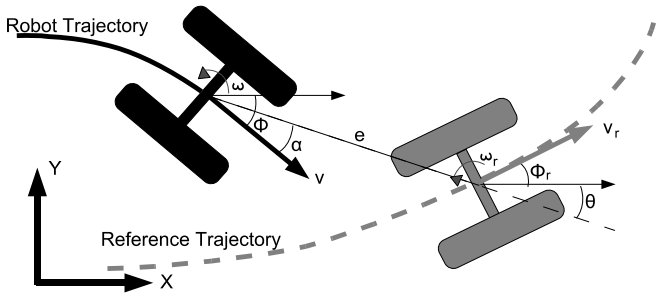

Fig. 2. Unicycle robot, Polar kinematics for trajectory tracking.

Communication networks have disturbances from time delays and lost data, resulting in varying sampling rates for control loops [3], [18]. This adversely effects the stability of motion controllers [8], [23]. In this research, we analyse the stability criterion of a state feedback controller over a network, and solve some of the greater issues with integrating this work into a complete teleoperated robot system.

III. Motion Control

For our system, we have chosen to adapt an existing kinematic state feedback controller from the literature [14], [24].

A. Kinematics

We use differentially steered wheeled mobile robots, which allows us to consider a unicycle kinematic model, as illustrated in Figure 2. A Cartesian to Polar state transformation is realized,

$$\begin{bmatrix} e \\ \theta \\ \alpha \end{bmatrix} = \begin{bmatrix} \sqrt{(x-x_r)^2+(y-y_r)^2} \\ atan2(-(y-y_r),-(x-x_r)) - \phi_r \\ \theta - \phi + \phi_r \end{bmatrix}, \quad (1)$$

where x, y, and ϕ are the Cartesian states, x_r, y_r and ϕ_r are the reference states, and e, θ, and α are the corresponding Polar states. Polar system state equations are defined,

$$\begin{bmatrix} \dot{e} \\ \dot{\theta} \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} v \cdot cos(\alpha) + v_r \cdot cos(\theta) \\ v \cdot \frac{sin(\alpha)}{e} - v_r \cdot \frac{sin(\alpha)}{e} - \dot{\phi}_r \\ v \cdot \frac{sin(\alpha)}{e} - v_r \cdot \frac{sin(\alpha)}{e} - \dot{\phi} \end{bmatrix}, \quad (2)$$

where v and v_r are the velocity and reference velocity respectively; and $\dot{\phi}$ and $\dot{\phi}_r$ are the rotational velocities. Equation (2) is solved in simulation, while in Mobile Emulab, (1) is used to calculate the Polar states.

B. Control Law

The nonlinear kinematic control laws used in this research were developed using Lyapunov based techniques [14]. This specific controller is capable of solving the posture stabilization, path following, and trajectory tracking problems

simultaneously. The optimized control law for linear velocity is given as,

$$v = \frac{\begin{pmatrix} k_1 \cdot e \cdot k_e \cdot tanh(e - r\sqrt{2} \cdot k_e) + \\ v_r \cdot e \cdot cos(\theta) \cdot k_e + \\ v_r \cdot k_r \cdot (sin(\theta) + \omega_r / v_r \cdot e) \end{pmatrix}}{e \cdot k_e + k_r \cdot sin(\alpha)}, \quad (3)$$

where r is the path manifold radius, k_1 controls the response, and k_e and k_r are defined as

$$k_e = \sqrt{\zeta - cos(2\theta)}, \quad (4)$$

where $\zeta = 1 - \epsilon$, and ϵ is a small perturbance to avoid a discontinuity at the origin; and

$$k_r = r\sqrt{2} \cdot sin(2\theta). \quad (5)$$

The optimized control law governing rotational velocity is given by,

$$\omega = k_2 \cdot tanh(\theta + \alpha) + 2\dot{\theta} + \dot{\phi_r}, \quad (6)$$

where k_2 is a gain to control the angular response of the controller.

C. Dynamic Extension

The dynamic extension [2], is defined by new states,

$$\dot{v_d} = -k_v(v_a - v_r) + \dot{v_r}, \quad (7)$$

$$\dot{\omega_d} = -k_\omega(\omega_a - \omega_r) + \dot{\omega_r}, \quad (8)$$

introduced to decrease steady state error and improve boundedness. This extension to the system also acts as a low pass filter, which improves the controller response in the presence of noisy state feedback. The gains k_v and k_ω are used to control the response of the dynamic extension. The variables v_r and ω_r are the values output from (3) and (6). The values v_a and ω_a are the measured robot velocity states. The states v_d and ω_d are the velocity commands sent to the robot.

IV. Methods

After rigorous simulation of the controller using MATLAB and SIMULINK, we implemented the full control system using the C programming language. The controller is implemented as its own independent set of functions, avoiding the need for changes to other parts of the system.

A. Program Structure

The implementation of the trajectory tracking controller is split into several components. The entire motion controller is in a single function called by the main robot control system. This function takes as input the current robot position sent by the localization system, the current data point on a parametric reference trajectory, and parameters related to the differentiated signals. Wheel velocities are returned as output, which are directly sent as commands to the robot.

Parameters and gains are then passed to the core controller, which uses the control laws (3) and (6). The resulting controller velocity commands are then sent to the dynamic extension, (7) and (8). The velocities v and ω are then transformed into wheel velocities v_L and v_R, and then sent to the robot over the network.

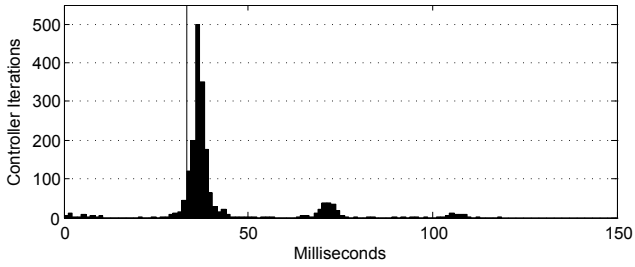

Fig. 3. Measured state feedback data sampling rate histogram.

B. System Parameters

The controller parameters must be properly tuned in order to assure stability and achieve the desired system performance. There are several design tradeoffs when considering controller parameters and gains. The response of the dynamic extension must be faster than the main controller response, or instability will result. If the time constants of the dynamic extension are too far below the sampling frequency of the motion controller implementation, the response will become unstable.

If the parameter ϵ is decreased, the controller will follow the path manifold [25] more aggressively. An increase of k_1 will cause the error, e to converge faster, and an increase in k_2 causes the controller to steer towards the path manifold more aggressively. Higher values of k_v and k_c increase the response of the dynamic extension, passing through v and ω with less filtering.

C. State Feedback Data Processing

We must solve several issues concerning data processing in order to assure stable robot trajectory tracking. Trigonometric functions used for converting to Polar coordinates must be multiphase in order to prevent discontinuities. A robust numerical differentiation and filtering method is needed to calculate state derivatives that can not be measured directly. Variability in sampling frequency, as illustrated in Figure 3, also presents challenges in the processing of state feedback data.

D. Stability Analysis

The trajectory tracking controller is modeled as a discrete system, and then analyzed to choose parameters that guarantee stability. Values for k_1 and k_2 are are chosen based on stability criterion, with constant values for r and ϵ. Values for k_v and k_ω are chosen based on further stability analysis considering k_1 and k_2.

We begin our stability analysis by defining error velocity states, given by $e_v = v_d - v_a$ and $e_\omega = \omega_d - \omega_a$; where the velocity states are as defined for (7) and (8). Solving for v_a and ω_a and substituting into (2) results in,

$$f(x) = \begin{bmatrix} -(e_v + v_d)\cos(\alpha) + v_r \cos(\theta) \\ \frac{(e_v + \omega_d)\sin(\alpha)}{e} - \frac{\omega_r \sin(\theta)}{e} - \omega_r \\ \frac{(e_v + v_d)\sin(\alpha)}{e} - \frac{\omega_r \sin(\theta)}{e} - e_w - \omega_d \end{bmatrix}. \quad (9)$$

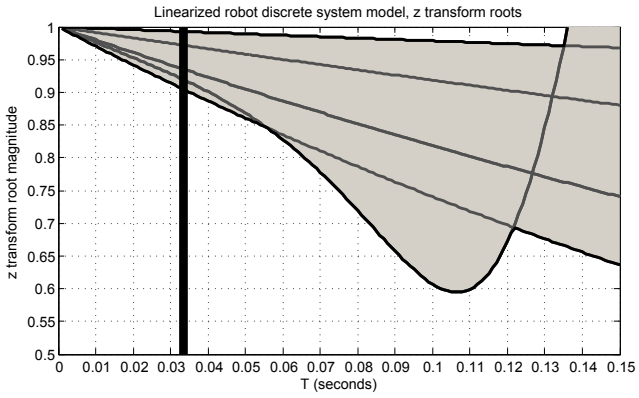

Fig. 4. Z transform roots with varying sample frequency.

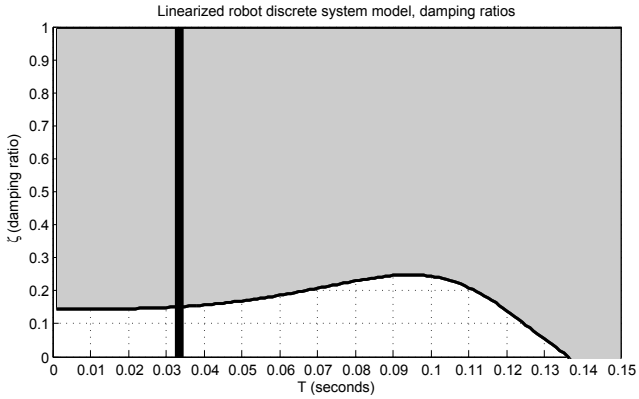

Fig. 5. Z transform damping ratios with varying sample frequency.

The nonlinear control laws (3) and (6) are then substituted into (9). The system equations for the robot are,

$$R = \begin{bmatrix} \dot{v}_a \\ \dot{\omega}_a \end{bmatrix} = \begin{bmatrix} -e_v/\tau_v \\ -e_\omega/\tau_\omega \end{bmatrix}, \tag{10}$$

where τ_v and τ_ω are the measured time constants of the robot for linear and rotational velocity, respectively. The full nonlinear system model of the robot becomes $G = [F; R]$.

An equilibrium point is chosen as $[e, \theta, \alpha, v, e_v, e_\omega] = [\epsilon, 0, 0, v_r, 0, 0]$. The nonlinear system is linearized by calculating the Jacobian about this equilibrium point. The linearized system is discussed in more detail in [6]. After linearization, we make the system discrete, by $L(z) = z * I - \phi$. The state transition matrix of L is calculated by a fourth order Taylor series approximation. System parameters are then substituted in to $L(z)$: A sampling frequency of $T = 1/30$, dynamic extension gains of $k_v = 3.0$, $k_c = 3.0$, controller gains of $k_1 = 0.85$, $k_2 = 0.5$, controller parameters of $\epsilon = 0.03$, $r = 0.2$, velocities of $v_r = 0.1$, $v = v_r$, and robot time constants of $\tau_v = 0.5$, $\tau_\omega = 0.5$. These values are determined by tuning the controller to achieve a desired system response, based on pole placement methods.

The desired root magnitude is calculated by, $Z = 10^{(Tlog(\tau))}$, where $T = 1/30$ seconds, and the desired time constant $\tau = 0.3$ seconds. This results in a root magnitude of $z = 0.9117$. We choose controller parameter values graphically to get as many roots as possible above this value, but below 1.0. Once controller parameters have been chosen, we iteratively tune the system based on performance data obtained through experimentation.

The z transform roots are obtained by solving $L(z) = 0$ for z. Figure 4 shows a plot of the z transform roots of our linearized system for varying sampling frequencies. The magnitude of all roots must be less than one for the system to meet stability criteria.

We obtain damping ratios by, $\zeta = -\frac{real(log(z)/T)}{|log(z)/T|}$. The corresponding damping ratios are given in Figure 5. The minimum damping ratio is approximately 0.2 for most sample frequencies. These low values correspond to the two robot states introduced in (10). The shaded area in each of these figures represents the range of values for the roots and damping ratios. Each figure shows seven different plots, corresponding to the seven states in the linearized discrete system.

By tuning the controller parameters based on the robot time constants and state feedback data frequency, the performance of the system was greatly improved. Higher controller velocity gains increase response, but may make the system unstable at low frequencies. Experimental evaluation is used to assist in obtaining the controller parameters.

V. Results

In this section, we present results of the controller in simulation, and experiments of the controller implemented into the testbed system. The effects of parameter tuning on this system are illustrated, and show the improvements caused by the discrete stability analysis. The curves used for the reference trajectories are constant radius circular arcs. There are curvature discontinuities at the boundaries of each curved segment.

A. Simulation

The controller is simulated at a sampling frequency corresponding to the expected frequency of localization data in the target system. The resulting trajectory calculated in this simulation is shown in Figure 6, with the corresponding system response in Figure 7. The disturbance from the discontinuity in curvature is apparent in the jumps in all three states at times $t \approx \{6, 9\}$.

B. Experiments

For our experiments, a single Acroname Garcia robot [7], is controlled by the Mobile Emulab system software. The experimental setup is identical to the environment discussed in [12]. A trajectory generated from a list of via points is sent to the motion controller, mimicking the data that would be received from the user. Each experiment is executed once, with the robot starting within 50 mm of the trajectory start point. The full Mobile Emulab system is utilized, including the overhead camera localization. Data from the localization system is recorded, and postprocessed to calculate the actual robot trajectory, velocities, and states.

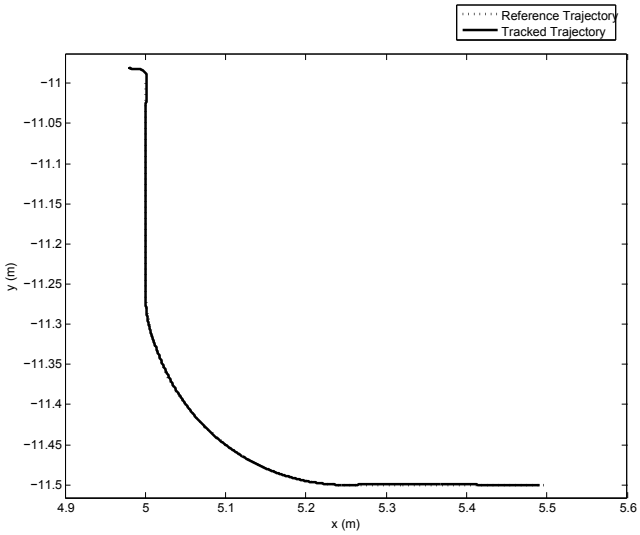

Fig. 6. Simulated trajectory with a single curve.

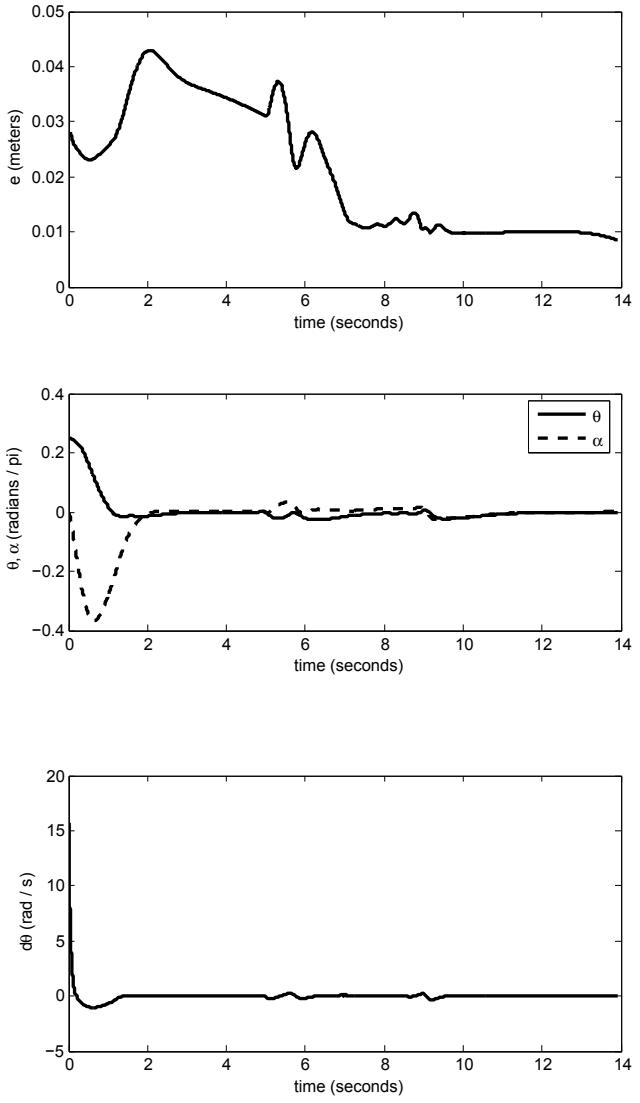

Fig. 7. System response of simulation given in Figure 6.

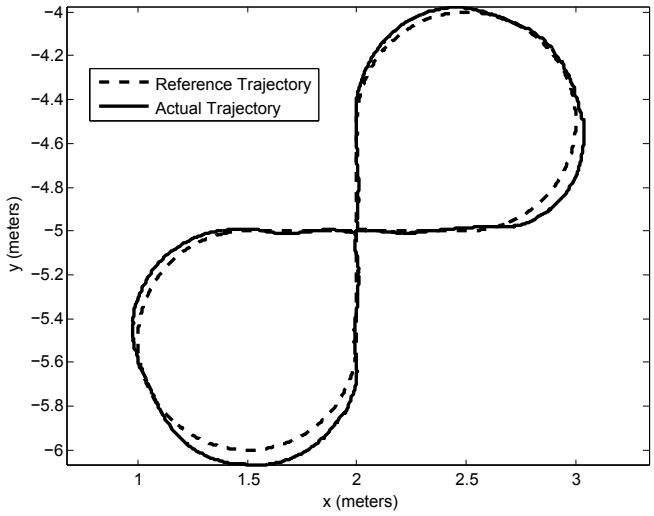

Fig. 8. Robot trajectory experiment, after parameter tuning.

We perform an experiment with controller parameters chosen based on the stability criteria discussed in Section IV-D. These values are $r = 0.02$ m, $\epsilon = 0.003$ m, $k_1 = 0.85$, $k_2 = 0.3$, and $k_v = k_\omega = 3.0$. The robot trajectory, as logged by the overhead camera localization system, is presented by Figure 8. The trajectory is closed, with an origin at $[2.0, -5.0]$ meters, with a heading of $\phi = 0.0$. The maximum configured reference velocity is $0.1m/s$, and the maximimum configured robot acceleration is $0.2m/s^2$.

The Polar states from (1) are plotted in Figure 9, logged directly from the motion control system.

C. Discussion

In all experiments, the robot starts with an initial position error of approximately ten millimeters. The controller used for motion control is designed for large initial position errors. Initial errors are expected to be very low, because robots are accurately positioned by the system prior to each experiment. Parametric trajectories consisting of line and arc segments are used. Initial and final velocities are zero, with a linear ramp function to constant velocity.

The robot is not capable of tracking the trajectory exactly at the boundaries between line and curved segments, because the change in curvature at this point is discontinuous. Oscillations present are caused by variability in the sampling rate of the controller. Noise and lag in the visual localization system further contribute to this problem.

The configurable wheel acceleration limits of the robots may be increased, but this also contributes to stability problems. A lower acceleration limit acts as a low pass filter, and helps reject disturbances in the velocity commands.

VI. Conclusion

In this article, we established the feasibility of robot trajectory tracking control running at a low sampling rate on a remote system. Future work includes the design and implementation of a discrete motion controller suited for this particular purpose, instead of adapting an existing

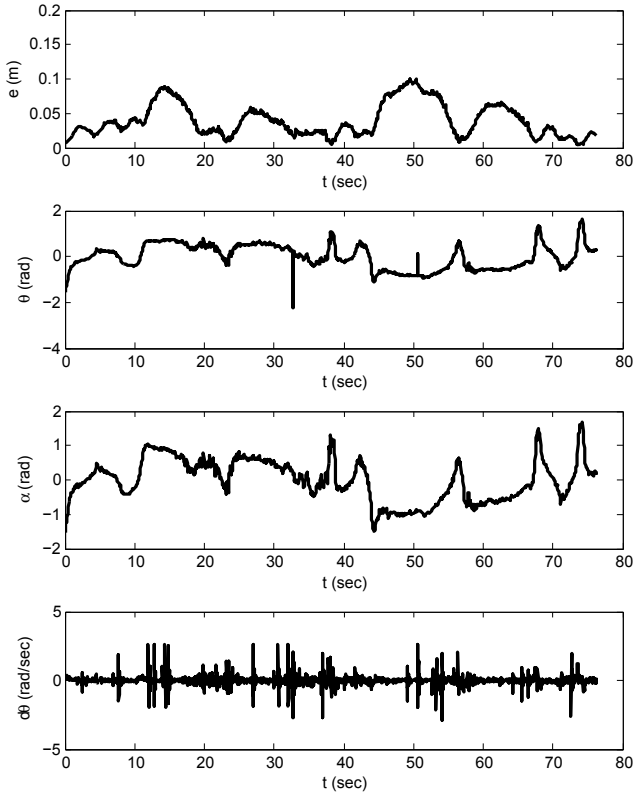

Fig. 9. System response corresponding to Figure 8.

controller. We tested a trajectory generator that produces more continuous paths in simulation. Cubic spirals and C^2 continuous splines were considered. These prospects would allow us to increase robot velocities, while decreasing tracking error. This would greatly benefit overall system performance, and enhance the robot capabilities presented to users and experimenters.

ACKNOWLEDGMENTS

The authors acknowledge the support and many contributions from the Flux Research Group in the School of Computing at the University of Utah. Of particular note are the contributions of David Johnson and Russ Fish with their work on the vision and robot localization systems. Thank you to Timothy Stack, Leigh Stoller, Kirk Webb, and Jay Lepreau for their many contributions to the Mobile Emulab system. This research was completed entirely at the University of Utah.

REFERENCES

[1] M. Aicardi, G. Casalino, A. Bicchi, and A. Balestrino, "Closed loop steering of unicycle like vehicles via lyapunov techniques," *IEEE Robot. Automat. Mag.*, vol. 2, no. 1, pp. 27 – 35, 1995.

[2] A. Bacciotti, "Local stabilizability of nonlinear control systems," *Series on Advances in Mathematics for Applied Sciences*, vol. 8, 1991.

[3] H. C. Cho and J. H. Park, "Stable bilateral teleoperation under a time delay using a robust impedance control," *Mechatronics*, vol. 15, no. 5, pp. 611 – 625, 2005.

[4] P. De, R. Krishnan, A. Raniwala, K. Tatavarthi, N. A. Syed, J. Modi, and T. cker Chiueh, "Mint-m: An autonomous mobile wireless experimentation platform," in *Mobisys*, 2006.

[5] R. Fierro and F. L. Lewis, "Control of a nonholonomic mobile robot using neural networks," *IEEE Trans. Neural Networks*, vol. 9, no. 4, pp. 589 – 600, 1998.

[6] D. M. Flickinger, "Motion planning and coordination of mobile robot behavior for medium scale distributed wireless network experiments," Master's thesis, University of Utah, May 2007, to be published.

[7] "Garcia robots from acroname, inc." [Online]. Available: http://-www.acroname.com/garcia/garcia.html

[8] F. Goktas, J. Smith, and R. Bajcsy, "Telerobotics over communication networks," in *36th IEEE Conference on Decision and Control*, vol. 3, San Diego, CA, USA, 1997, pp. 2399 – 2404.

[9] K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, and J. Wiegley, "Desktop teleoperation via the world wide web," in *IEEE International Conference on Robotics and Automation*, vol. 1, Nagoya, Jpn, 1995, pp. 654 – 659.

[10] M.-Y. Hsieh, V. Kumar, and C. Taylor, "Constructing radio signal strength maps with multiple robots," in *IEEE International Conference on Robotics and Automation*, vol. 4, 2004, pp. 4184 – 4189.

[11] G. Indiveri, "Kinematic time-invariant control of a 2d nonholonomic vehicle," in *1999 Conference on Decision and Control*, vol. 3, 1999, pp. 2112 – 2117.

[12] D. Johnson, T. Stack, D. M. Flickinger, L. Stoller, R. Ricci, and J. Lepreau, "Mobile emulab: A robotic wireless and sensor network testbed," in *the 25th Conference on Computer Communications (IEEE INFOCOM 2006)*, Barcelona, Spain, Apr. 2006.

[13] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," in *IEEE International Conference on Robotics and Automation*, 1990, pp. 384 – 389.

[14] Y. Kim and M. A. Minor, "Path manifold based kinematic control of wheeled mobile robots considering physical constraints," *Int. Journal of Robotics Research*, 2007, under revision.

[15] I. Kolmanovsky and N. McClamroch, "Developments in nonholonomic control problems," *IEEE Control Syst. Mag.*, vol. 15, no. 6, pp. 20 – 36, Dec. 1995.

[16] D. Lee, O. Martinez-Palafox, and M. W. Spong, "Bilateral teleoperation of a wheeled mobile robot over delayed communication network," in *IEEE International Conference on Robotics and Automation*, Orlando, FL, United States, 2006, pp. 3298 – 3303.

[17] R. Luo, K. Su, S. Shen, and K. Tsai, "Networked intelligent robots through the internet: issues and opportunities," in *IEEE*, vol. 91, no. 3, Mar. 2003, pp. 371 – 382.

[18] J. H. Park and H. C. Cho, "Sliding-mode controller for bilateral teleoperation with varying time delay," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Atlanta, GA, USA, 1999, pp. 311 – 316.

[19] N. Sgouros and S. Gerogiannakis, "Robot teleoperation environments featuring wap-based wireless devices," *J. Netw. Comput. Appl. (UK)*, vol. 26, no. 3, pp. 259 – 271, Aug. 2003.

[20] R. Simmons, J. Fernandez, R. Goodwin, S. Koenig, and J. O'Sullivan, "Lessons learned from xavier," *IEEE Robot. Automat. Mag.*, vol. 7, no. 2, pp. 33 – 39, June 2000.

[21] E. A. Thompson, E. Harmison, R. Carper, R. Martin, and J. Isaacs, "Robot teleoperation featuring commercially available wireless network cards," *Journal of Network and Computer Applications*, vol. 29, no. 1, pp. 11 – 24, 2006.

[22] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *Fifth Symposium on Operating Systems Design and Implementation*, Boston, MA, Dec. 2002, pp. 255 – 270.

[23] S. Yadlapalli, S. Darbha, and K. Rajagopal, "Information flow and its relation to stability of the motion of vehicles in a rigid formation," *IEEE Trans. Autom. Control (USA)*, vol. 51, no. 8, pp. 1315 – 1319, Aug. 2006.

[24] X. Zhu, Y. Kim, and M. A. Minor, "Cooperative distributed robust control of modular mobile robots with bounded curvature and velocity," in *2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Monterey, California, 2005.

[25] X. Zhu, M. A. Minor, and S. Park, "Distributed robust control of compliant framed wheeled modular mobile robots," *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 128, no. 3, pp. 489 – 498, 2006.