Vietnam Journal of Mechanics, VAST, Vol. 36, No. 3 (2014), pp. 215 – 233

STABLE CONTROL OF NETWORKED ROBOT SUBJECT TO COMMUNICATION DELAY, PACKET LOSS, AND OUT-OF-ORDER DELIVERY

Manh Duong Phung*, Thuan Hoang Tran, Quang Vinh Tran
University of Engineering and Technology, VNU, Hanoi, Vietnam
*E-mail: duongpm@vnu.edu.vn
Received July 02, 2013

Abstract. Stabilization control of networked robot system faces uncertain factors caused by the network. Our approach for this problem consists of two steps. First, the Lyapunov stability theory is employed to derive control laws that stabilize the non-networked robot system. Those control laws are then extended to the networked robot system by implementing a predictive filter between the sensor and controller. The filter compensates influences of the network to acquire accurate estimate of the system state and consequently ensures the convergence of the control laws. The optimality of the filter in term of minimizing the mean square error is theoretically proven. Many simulations and experiments have been conducted. The result confirmed the validity of the proposed approach.

Keywords: Networked robot, stabilization control, optimal filter

1. INTRODUCTION

The stable movement from one point to another is essential for the efficient operation of a control system and is basic for the development of real-world applications. In non-networked robot system, a number of researches have been introduced and the problem of stabilization control has been solved in both theoretic and experimental aspects [1–3]. Networked robot systems (NRSs) however have differences. The occurrence of network delay, packet loss and out-of-order delivery influences the accuracy of state estimation and control so that directly applying previous control methods is no longer practical. Several new approaches have been proposed.

Wargui et al. developed a stable controller for NRSs with nonholonomic constraints [4]. Control laws that stabilize the system in delay-free scenario are first derived. They are then extended to work with the time delay by adding a state estimator between the system output and the controller. The estimator uses a multistep prediction mechanism to predict the state that will be used for the controller. In [5], the maximum time delay allowed at the control input without the loss of system stability is estimated. Based on it, a single layered neural network is designed for the controller to control the robot with unknown dynamics. In [6], Luck uses a time buffer which was longer than the worst delay

to make the system to be time-invariant and then applied the classic control theory. In [7], Xi and Tarn propose an event-based (non-time) control scheme to reduce the impact of time delay on the system operation. This idea is expanded by Wang and Liu [8] in which they introduced the telecommanding concept. In telecommanding, each command is designed for an independent task and is defined with multiple events so that the robot can deliberately respond to expected events while actively respond to unexpected events. The main limitation of the proposed works is that they mainly focus on dealing with the time delay, hardly address other issues such as the out-of-order delivery.

In this study, we address the problem of stabilization of NRS under the influence of three inevitable network induced problems: the communication delay, packet loss, and out-of-order delivery. The core of the scheme is the development of a new filter that is able to predict the robot's pose based on past observations so that control laws designed for non-networked robot system can be used to stabilize the NRS. The filter is optimal in sense that it minimizes the mean of the squared error. In addition, expansion of the filter to nonlinear systems is also derived. The paper is organized as follows. Section 2 introduces the system model and formulates the control problem. Section 3 describes the stabilization of non-networked robot system. The stabilization of NRS is introduced in section 4. Section 5 presents the simulations, experiments, and discussions. Conclusions are drawn in section 4.

2. SYSTEM MODEL AND PROBLEM FORMULATION

2.1. System Model

The robot considered in this study is the two-wheeled, differential-drive mobile robot with non-slipping and pure rolling. The pose or state of the robot includes the position of the wheel axis center (x, y) and the chassis orientation θ with respect to the x axis (Fig. 1).

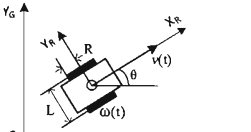


Fig. 1. The robot's pose and parameters

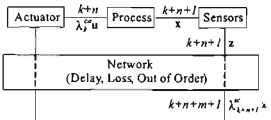


Fig. 2. Model of a networked robot system

The kinematic models of the robot in continuous and discrete time domains in case of no noise affected are given by

$$\dot{x} = v\cos\theta, \quad \dot{y} = v\sin\theta, \quad \dot{\theta} = \omega, \tag{1}$$

$$x_{k+1} = x_k + T_s v_k \cos\theta_k, \; y_{k+1} = y_k + T_s v_k \sin\theta_k, \; \theta_{k+1} = \theta_k + T_s\omega_k, \tag{2}$$

where T_s is the sampling period and v and ω are the tangent and angular velocities of the robot, respectively. In practice, there always exist two kinds of noises in a robotic system including the input and measurement noises. In this study, these noises are assumed to be independent, zero-mean, and white-noise processes with normal probability distributions: $\mathbf{w}_k \sim \mathbf{N}(0, Q_k)$, $\mathbf{v}_k \sim \mathbf{N}(0, R_k)$, $E(\mathbf{w}_i \mathbf{v}_j^T) = 0$. This assumption is sufficient and widely adopted due to the central limit theorem [9–12]. With the existence of noises, the robot system can be described in state-space representation as follows.

Let $\mathbf{x} = [x\ y\ \theta]^T$ be the state vector. This state can be observed by a measurement $\mathbf{z}$. This measurement is described by a function h of the state vector and measurement noise $\mathbf{v}$. Denoting function (2) as f, with an input vector $\mathbf{u} = [v\ \omega]$ and input disturbance $\mathbf{w}$, the representation of the robot in state space is given by

$$\begin{aligned} \mathbf{x}_{k+1} &= f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k), \\ \mathbf{z}_k &= h(\mathbf{x}_k, \mathbf{v}_k). \end{aligned} \tag{3}$$

When distributing over communication networks, the robot becomes decentralized and its functioning operation depends on a number of network parameters. The networks are in general very complex and can greatly differ in their architecture and implementation depending on the medium used, and on the applications they are meant to serve. In this work, a network is modeled as a module between the process and controller which delivers input signals and observation measurements with possible delay, loss, and out-of-order (Fig. 2). The delay is assumed to be random but measurable at each sampling time. The packet loss is modeled as a binary random variable λ_k defined as follows

$$\lambda_k = \begin{cases} 1, & \text{if a packet arrives at time } k \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The out-of-order packet with sequence number i arrived at time $k(i < k)$ is modeled as a delayed packet with the time delay

$$\Delta t_i = \Delta t_k + (j - i)T_s, \tag{5}$$

where Δt_k is the transfer time at time k and j is the sequence number of the last received chronological packet.

Let n be the network delay (in number of sampling periods) between the controller and the actuator, m be the network delay between the sensor and the controller, λ_k^{ca} be the binary random variable described the arrival of inputs from the controller to the actuator, λ_k^{sc} be the binary random variable described the arrival of measurements from the sensor to the controller. The system (3) becomes time-varying and can be rewritten as follows

$$\begin{aligned} \mathbf{x}_k &= f(\mathbf{x}_{k-1}, \lambda_{k-n-1}^{ca} \mathbf{u}_{k-n-1}, \mathbf{w}_{k-1}), \\ \tilde{\mathbf{z}}_k &= \lambda_{k-m}^{sc} \mathbf{z}_{k-m} = \lambda_{k-m}^{sc} h(\mathbf{x}_{k-m}, \mathbf{v}_{k-m}). \end{aligned} \tag{6}$$

2.2. Problem Formulation

Consider the control scenario shown in Fig. 3, with an arbitrary position and orientation of the robot and a predefined goal position and orientation. Let the difference between the present pose (x, y, θ) and the final pose (x_2, y_2, θ_2) given in the robot reference frame $\{X_R, Y_R, \theta_R\}$ be the error vector, $\mathbf{e} = (x_2 - x, y_2 - y, \theta_2 - \theta)^T$. The task of

the controller layout is to find a control constraint, if it exists, of the tangent and angular velocities such that the error e is driven toward zero: $\lim_{t\to\infty} \mathbf{e}(t) = 0$.

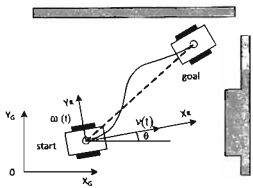


Fig. 3. The goal of the controller

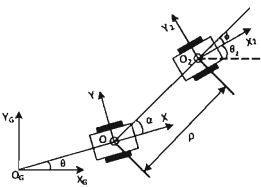


Fig. 4. The robot poses and navigation variables

Our approach for this problem consists of two steps. First, control laws that stabilize the non-networked robot system are derived. Then, a predictive filter is introduced to extend those control laws to the NRS.

3. STABILIZATION OF NON-NETWORKED ROBOT SYSTEM

In (1), it can be seen that one degree of freedom of the system is constrained as

$$\dot{x}\sin\theta - \dot{y}\cos\theta = 0. \tag{7}$$

Thus, the system is nonholonomic. According to a work of Brockett [13], Cartesian state-space representations of the robot cannot be asymptotically stabilized via smooth and time invariant feedback laws. We therefore define a new coordinate system as shown in Fig. 4.

The new coordinate system consists of three parameters (ρ, α, ϕ) called navigation variables. Let OXY and $O_2X_2Y_2$ be the local coordinate frames attached to the present and final poses of the robot, respectively, ρ is then defined as the distance between O and O_2, ϕ is the angular made by the vector connecting O and O_2 and the vector connecting O_2 and x_2, and α is the angular made by the vector connecting O and O_2 and the vector connecting O and x

$$\begin{aligned} \rho &= \sqrt{(x_2 - x)^2 + (y_2 - y)^2}, \\ \varphi &= a\tan 2\,(y_2 - y, x_2 - x) - \theta_2, \\ \alpha &= a\tan 2\,(y_2 - y, y_2 - x) - \theta. \end{aligned} \tag{8}$$

Without lost of generality, we assume that the goal desired pose of the robot is $(x_2, y_2, \theta_2) = (0, 0, 0)$ which can also be expressed by $(\rho_2, \alpha_2, \varphi_2) = (0, 0, 0)$ (This can be done by setting the origin of the global coordinate frame at the final position). Let w_v and w_ω be the input disturbances of the tangent and angular velocities, respectively. With the existing of input disturbances, the kinematic model (1) is rewritten in navigation variable

domain (ρ, α, ϕ) as follows

$$\begin{aligned} \dot{\rho} &= -(v + w_v)\cos\alpha, \\ \dot{\alpha} &= -(\omega + w_\omega) + (v + w_v)\frac{\sin\alpha}{\rho}, \\ \dot{\varphi} &= (v + w_\omega)\frac{\sin\alpha}{\rho}. \end{aligned} \tag{9}$$

The goal of the controller now is to establish smooth control laws that drive the navigation variables (ρ, α, ϕ) toward zero.

By using Lyapunov theory, it is shown that (9) can be asymptotically stabilized using the following control laws

$$\begin{aligned} v &= (\gamma\cos\alpha)e, \\ \omega &= \lambda\alpha + \gamma\frac{\cos\alpha\sin\alpha}{\alpha}(\alpha + h\theta). \end{aligned} \tag{10}$$

where γ, λ and h are the positive parameters [1, 14]. In discrete time domain, the control laws are given by

$$\begin{aligned} v_k &= (\gamma\cos\alpha_k)e_k, \\ \omega_k &= \lambda\alpha_k + \gamma\frac{\cos\alpha_k\sin\alpha_k}{\alpha_k}(\alpha_k + h\theta_k). \end{aligned} \tag{11}$$

Those control laws combined with a predictive filter are capable to stabilize the NRS as presented in the next section.

4. STABILIZATION OF NETWORKED ROBOT SYSTEM

As shown in (6), the NRS is time-varying in which the control input at time k would not reach the actuator until time $k + n$ while the measurement at time k actually reflects the system state at time $k - m$. Thus, in order to ensure the stabilization of control laws (11) for NRS, the system state at time $k + n$ need be estimated based on measurements taken at time $k - m$, $\hat{\mathbf{x}}(k + n|k - m)$ (Fig. 5). This estimate will be used at the controller to generate the control signal $\mathbf{u}_{k+n}$ so that it will arrive to the actuator at time $k + n$ as expected. This approach is similar to [4] but we have developed a new state estimator which is able to handle not only the network delay but also the packet loss and out-of-order delivery. The filter called *past observation-based predictive filter* (PO-PF) is derived based on the Kalman filter's theory [9].

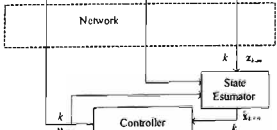


Fig. 5. NRS with the presence of a state estimator

In this section, the standard Kalman filter is first briefly described. The derivation of the PO-PF for linear systems is then introduced. Finally, the expansion of the PO-PF for nonlinear systems is presented.

4.1. The standard Kalman filter

Consider the following discrete time linear stochastic system

$$\begin{aligned} \mathbf{x}_k &= A_{k-1}\mathbf{x}_{k-1} + B_{k-1}\mathbf{u}_{k-1} + \mathbf{w}_{k-1}, \\ \mathbf{z}_k &= H_k\mathbf{x}_k + \mathbf{v}_k, \end{aligned} \tag{12}$$

where $k \in \mathrm{N}$, $\mathbf{x}$ and $\mathbf{w} \in \mathrm{R}^n$, $\mathbf{z}$ and $\mathbf{v} \in \mathrm{R}^m$, $\mathbf{u} \in \mathrm{R}^l$, $A \in \mathrm{R}^{n\times n}$, $B \in \mathrm{R}^{n\times l}$, $H \in \mathrm{R}^{m\times n}$, $(\mathbf{x}_0, \mathbf{w}, \mathbf{v})$ are Gaussian, uncorrelated, white, with mean $(\bar{\mathbf{x}}, 0, 0)$ and covariance (P_0, Q, R) respectively. The steps to calculate the Kalman filter can be summarized as follows [9]:

The time update equations (prediction phase)

$$\hat{\mathbf{x}}_k^- = A_{k-1}\hat{\mathbf{x}}_{k-1}^+ + B_{k-1}\mathbf{u}_{k-1}, \tag{13}$$

$$P_k^- = A_{k-1}P_{k-1}^+A_{k-1}^T + Q_{k-1}, \tag{14}$$

where $\hat{\mathbf{x}}_k^- \in \mathrm{R}^n$ is the *priori* state estimate at step k given knowledge of the process prior to step k, and P_k^- denotes the covariance matrix of the *priori* estimate error.

The data update equations (correction phase)

$$K_k = P_k^- H_k^T[H_kP_k^-H_k^T + R_k]^{-1}, \tag{15}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + K_k[\mathbf{z}_k - H_k\hat{\mathbf{x}}_k^-], \tag{16}$$

$$P_k^+ = [I - K_kH_k]P_k^-, \tag{17}$$

where $\hat{\mathbf{x}}_k^+ \in \mathrm{R}^n$ is the *posteriori* state estimate at step k given measurement $\mathbf{z}_k$, K_k is the Kalman gain, and P_k^+ is the covariance matrix of the *posteriori* estimate error.

4.2. The past observation-based predictive filter for linear NRSs

Consider the NRS described in (6). If functions f and h are linear, the system is rewritten as

$$\begin{aligned} \mathbf{x}_k &= A_{k-1}\mathbf{x}_{k-1} + \lambda_{k-n-1}^{ca}B_{k-1}\mathbf{u}_{k-n-1} + \mathbf{w}_{k-1} \\ &= A_{k-1}\mathbf{x}_{k-1} + B_{k-1}\tilde{\mathbf{u}}_{k-n-1} + \mathbf{w}_{k-1}, \end{aligned} \tag{18}$$

$$\begin{aligned} \tilde{\mathbf{z}}_k^i &= \lambda_{k-m}^{sc}\mathbf{z}_{k-m} \\ &= \lambda_{k-m}^{sc}H_{k-m}\mathbf{x}_{k-m} + \lambda_{k-m}^{sc}\mathbf{v}_{k-m} \\ &= \tilde{H}_i\mathbf{x}_i + \tilde{\mathbf{v}}_i, \end{aligned} \tag{19}$$

where $\tilde{\mathbf{u}}_{k-n-1}$, $\tilde{\mathbf{z}}_k^i$, $\tilde{H}_{k-m}$, $\tilde{\mathbf{v}}_{k-m}$, and i are defined by the above equations. We can derive the optimal filter to estimate the state of the stochastic linear system given in (18) and (19) as follows.

Priori State Estimate at prediction phase

The *priori* estimate, $\hat{\mathbf{x}}_k^-$, is defined as the expectation of the state $\mathbf{x}_k$ given all measurements up to and including the last step $k-1$. From (18), it is given by

$$\hat{\mathbf{x}}_k^- = E(\mathbf{x}_k) = A_{k-1}E(\mathbf{x}_{k-1}) + B_{k-1}E(\tilde{\mathbf{u}}_{k-n-1}) + E(\mathbf{w}_{k-1}). \tag{20}$$

As $E(\mathbf{x}_{k-1})$ is the *posteriori* state estimate at time $k-1$, $\tilde{\mathbf{u}}_{k-n-1}$ is a known input, and $\mathbf{w}_{k-1}$ is zero-mean, (20) becomes

$$\hat{\mathbf{x}}_k^- = A_{k-1}\hat{\mathbf{x}}_{k-1}^+ + B_{k-1}\tilde{\mathbf{u}}_{k-n-1}. \tag{21}$$

At time k, the controller sends the input signal $\tilde{\mathbf{u}}_{k+n}$, (21) becomes

$$\hat{\mathbf{x}}_k^- = A_{k-1}\hat{\mathbf{x}}_{k-1}^+ + B_{k-1}\tilde{\mathbf{u}}_{k-1}. \tag{22}$$

Priori Error Covariance

Let $\mathbf{e}_k^-$ and $\mathbf{e}_k^+$ be the *priori* and *posteriori* estimate errors, respectively

$$\mathbf{e}_k^- = \mathbf{x}_k - \hat{\mathbf{x}}_k^-, \tag{23}$$

$$\mathbf{e}_k^+ = \mathbf{x}_k - \hat{\mathbf{x}}_k^+. \tag{24}$$

From (18) and (22), the covariance of the *priori* estimate error is given by

$$\begin{aligned} P_k^- &= E(\mathbf{e}_k^-\mathbf{e}_k^{-T}) \\ &= E(A_{k-1}\mathbf{e}_{k-1}^+\mathbf{e}_{k-1}^{+T}A_{k-1}^T + \mathbf{w}_{k-1}\mathbf{w}_{k-1}^T \\ &\quad + A_{k-1}\mathbf{e}_{k-1}^+\mathbf{w}_{k-1}^T + \mathbf{w}_{k-1}\mathbf{e}_{k-1}^{+T}A_{k-1}^T) \\ &= A_{k-1}P_{k-1}^+A_{k-1}^T + Q_{k-1}. \end{aligned} \tag{25}$$

Posteriori State Estimate at correction phase

From (19) the measurement $\tilde{\mathbf{z}}_k^i$ received at time k would update the system state at a previous time i rather than the present time k. Due to the network, this measurement could not reach the estimator until time k. We therefore construct the data update equation of the form

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + K_k(\tilde{\mathbf{z}}_k^i - \tilde{H}_i\hat{\mathbf{x}}_i^-), \tag{26}$$

and recompute the Kalman gain and error covariance to ensure the optimality of the new data update equation.

Kalman gain and Posteriori Error Covariance

Assume that the measurement is fused using (26) with an arbitrary gain K_k. The covariance of the *posteriori* estimate error, P_k^+, is determined as

$$\begin{aligned} P_k^+ &= E(\mathbf{e}_k^+\mathbf{e}_k^{+T}) \\ &= E[\mathbf{e}_k^-\mathbf{e}_k^{-T} - \mathbf{e}_k^-\mathbf{e}_i^{-T}(K_k\tilde{H}_i)^T - K_k\tilde{H}_i\mathbf{e}_i^-\mathbf{e}_k^{-T} \\ &\quad + K_k\tilde{H}_i\mathbf{e}_i^-\mathbf{e}_i^{-T}(K_k\tilde{H}_i)^T + K_k\tilde{H}_i\mathbf{e}_i^-\tilde{\mathbf{v}}_i^TK_k^T - K_k\tilde{\mathbf{v}}_i\mathbf{e}_k^{-T} \\ &\quad + K_k\tilde{\mathbf{v}}_i\mathbf{e}_i^{-T}(K_k\tilde{H}_i)^T + K_k\tilde{\mathbf{v}}_i\tilde{\mathbf{v}}_i^TK_k^T]. \end{aligned} \tag{27}$$

Due to the independence between $\mathbf{e}^-$ and $\tilde{\mathbf{v}}$, (27) can be simplified to

$$\begin{aligned} P_k^+ &= E(\mathbf{e}_k^- \mathbf{e}_k^{-T}) - E(\mathbf{e}_k^- \mathbf{e}_i^{-T})(K_k \tilde{H}_i)^T - K_k \tilde{H}_i E(\mathbf{e}_i^- \mathbf{e}_k^{-T}) \\ &\quad + K_k \tilde{H}_i E(\mathbf{e}_i^- \mathbf{e}_i^{-T})(K_k \tilde{H}_i)^T + K_k E(\tilde{\mathbf{v}}_i \tilde{\mathbf{v}}_i^T) K_k^T] \\ &= P_k^- - L^T \tilde{H}_i^T K_k^T - K_k \tilde{H}_i L + K_k \tilde{H}_i P_i^- \tilde{H}_i^T K_k^T + K_k \tilde{R}_i K_k^T, \end{aligned} \tag{28}$$

where $L = E(\mathbf{e}_i^- \mathbf{e}_k^{-T})$.

As the matrix K_k is chosen to be the gain or blending factor that minimizes the *posteriori* error covariance, this minimization is accomplished by taking the derivative of the trace of the *posteriori* error covariance with respect to K_k, setting that result equal to zero, and then solving for K_k. Applying this process to (28) obtains

$$\frac{\partial tr(P_k^+)}{\partial K_k} = 2(-L^T \tilde{H}_i^T + K_k \tilde{H}_i P_i \tilde{H}_i^T + K_k \tilde{R}_i) = 0, \tag{29}$$

$$\Leftrightarrow K_k = L^T \tilde{H}_i^T [\tilde{H}_i P_i^- \tilde{H}_i^T + \tilde{R}_i]^{-1} \tag{30}$$

Inserting (30) in (28) leads to a simpler form of P_k^+

$$P_k^+ = P_k^- - K_k \tilde{H}_i L. \tag{31}$$

In order to compute L, the *priori* state estimate at time k needs determining from the estimate at time i. Through the time update (21) and the data update (26), $\mathbf{e}^-$ becomes

$$\begin{aligned} \mathbf{e}_k^- &= \mathbf{x}_k - \hat{\mathbf{x}}_k^- \\ &= A_{k-1} \mathbf{e}_{k-1}^+ - \mathbf{w}_{k-1} \\ &= A_{k-1}[(I - K_{k-1} \tilde{H}_{k-1}) \mathbf{e}_{k-1}^- + K_{k-1} \tilde{\mathbf{v}}_{k-1}] - \mathbf{w}_{k-1}. \end{aligned} \tag{32}$$

After m updating steps, the estimation error becomes

$$\mathbf{e}_k^- = F \mathbf{e}_i^- + \xi_1(\mathbf{w}_i, \ldots, \mathbf{w}_{k-1}) + \xi_2(\tilde{\mathbf{v}}_i, \ldots, \tilde{\mathbf{v}}_{k-1}), \tag{33}$$

where

$$F = \prod_{j=1}^{m} A_{k-j}(I - K_{k-j} \tilde{H}_{k-j}), \tag{34}$$

and ξ_1 and ξ_2 are the functions of noise sequences $\mathbf{w}$ and $\tilde{\mathbf{v}}$. From (33) and the independence between $\mathbf{e}^-$ and noise sequences, the covariance L becomes

$$L = E(\mathbf{e}_i^- \mathbf{e}_k^{-T}) = P_i^- F^T. \tag{35}$$

Substituting (35) in (31) and (30) yields

$$P_k^+ = P_k^- - K_k \tilde{H}_i P_i^- F^T, \tag{36}$$

and

$$K_k = F P_i^- \tilde{H}_i^T [\tilde{H}_i P_i^- \tilde{H}_i^T + \tilde{R}_i]^{-1} \tag{37}$$

Predictive state estimate at extrapolated phase

The extrapolated phase is added to predict the system state at time $k+n$ from time k. As there is no additional measurement, this phase is an open-loop propagation

$$\hat{\mathbf{x}}_{k+n}^{-} = A_{k+n-1}\hat{\mathbf{x}}_{k+n-1}^{+} + B_{k+n-1}\tilde{\mathbf{u}}_{k+n-1}. \tag{38}$$

- Remark 1: When the delay n and m is zero, $F = I$ and the Kalman gain (37) reduces to the standard form (15) and the error covariance (36) reduces to (17)
- Remark 2: Eq. (37) can be rewritten as

$$K_k = FK_i^{\star}, \tag{39}$$

where $K_i^{\star}$ is the Kalman gain at time i of the standard Kalman filter (15). It turns out that the past residual $(\tilde{\mathbf{z}}_k^{-} - \tilde{H}_i\hat{\mathbf{x}}_i^{-})$ in Eq. (26) can be normally updated to the *posterior* estimate at time k as at time i but the Kalman gain needs to be changed by the factor F. This factor describes the relevant of the measurement updated at time i to the state at time k.

- Remark 3: Eq. (26) can be rewritten as

$$\hat{\mathbf{x}}_k^{+} = \hat{\mathbf{x}}_k^{-} + \lambda_i K_k(\mathbf{z}_k^{i} - H_i\hat{\mathbf{x}}_i^{-}). \tag{40}$$

It implies that if a measurement is not arrived ($\lambda_i = 0$), the estimator does not use any information of the "dummy" observation to the estimate It simply sets the *posteriori* estimate to the value of the *priori* estimate.

4.3. The past observation-based predictive filter for nonlinear NRSs

Though the filter derived in previous section is capable for NRSs, the system has to be linear. As practical robot systems are often nonlinear, further modification needs to be accomplished. The main idea is the linearization of a nonlinear system around its previous estimated states.

Performing a Taylor series expansion of the state equation around $(\hat{\mathbf{x}}_{k-1}^{+}, \tilde{\mathbf{u}}_{k-1}, 0)$ gives

$$\begin{aligned}
\mathbf{x}_k &= f(\hat{\mathbf{x}}_{k-1}^{+}, \tilde{\mathbf{u}}_{k-1}, 0) + \frac{\partial f}{\partial \mathbf{x}}\Big|_{(\hat{\mathbf{x}}_{k-1}^{+}, \mathbf{u}_{k-1}, 0)} (\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}^{+}) + \frac{\partial f}{\partial \mathbf{w}}\Big|_{(\hat{\mathbf{x}}_{k-1}^{+}, \mathbf{u}_{k-1}, 0)} \mathbf{w}_{k-1} \\
&= f(\hat{\mathbf{x}}_{k-1}^{+}, \tilde{\mathbf{u}}_{k-1}, 0) + A_{k-1}(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}^{+}) + W_{k-1}\mathbf{w}_{k-1} \\
&= A_{k-1}\mathbf{x}_{k-1} + [f(\hat{\mathbf{x}}_{k-1}^{+}, \tilde{\mathbf{u}}_{k-1}, 0) - A_{k-1}\hat{\mathbf{x}}_{k-1}^{+}] + W_{k-1}\mathbf{w}_{k-1} \\
&= A_{k-1}\mathbf{x}_{k-1} + \tilde{\mathbf{u}}_{k-1}^{\star} + \mathbf{w}_{k-1}^{\star},
\end{aligned} \tag{41}$$

where A_{k-1}, W_{k-1}, $\tilde{\mathbf{u}}_{k-1}^{\star}$, $\tilde{\mathbf{w}}_{k-1}^{\star}$ are defined by the above equation. Similarly, the measurement equation is linearized around $(\hat{\mathbf{x}}_i^{-}, 0)$ to obtain

$$\begin{aligned}
\tilde{\mathbf{z}}_k^{i} &= \lambda_i[h(\hat{\mathbf{x}}_i^{-}, 0) + \frac{\partial h}{\partial \mathbf{x}}\Big|_{(\hat{\mathbf{x}}_i^{-}, 0)} (\mathbf{x}_i - \hat{\mathbf{x}}_i^{-}) + \frac{\partial h}{\partial \mathbf{v}}\Big|_{(\hat{\mathbf{x}}_i^{-}, 0)} \mathbf{v}_i] \\
&= \tilde{h}(\hat{\mathbf{x}}_i^{-}, 0) + \tilde{H}_i(\mathbf{x}_i - \hat{\mathbf{x}}_i^{-}) + \tilde{V}_i\mathbf{v}_i \\
&= \tilde{H}_i\mathbf{x}_i + [\tilde{h}(\hat{\mathbf{x}}_i^{-}, 0) - \tilde{H}_i\hat{\mathbf{x}}_i^{-}] + \tilde{V}_i\mathbf{v}_i \\
&= \tilde{H}_i\mathbf{x}_i + \varepsilon_i^{\star} + \tilde{\mathbf{v}}_i^{\star},
\end{aligned} \tag{42}$$

where $\tilde{h}_i$, $\tilde{H}_i$, $\tilde{V}_i$, ε_i^x, $\tilde{\mathbf{v}}_i^x$ are defined by the above equation. The system (41) and the measurement (42) are now linear and the proposed filter can be applied to obtain the optimal filter for nonlinear NRSs as follows:

- The time update equations at prediction phase

$$\begin{aligned} \hat{\mathbf{x}}_k^- &= f_{k-1}(\hat{\mathbf{x}}_{k-1}^+, \tilde{\mathbf{u}}_{k-1}, \mathbf{0}), \\ P_k^- &= A_{k-1} P_{k-1}^+ A_{k-1}^T + W_{k-1} Q_{k-1} W_{k-1}^T. \end{aligned} \tag{43}$$

- The data update equations at correction phase

$$\begin{aligned} F &= \prod_{j=1}^{m} A_{k-j}(I - K_{k-j}\tilde{H}_{k-j}), \\ K_k &= F P_i^- \tilde{H}_i^T (\tilde{H}_i P_i^- \tilde{H}_i^T + \tilde{V}_i \tilde{R}_i \tilde{V}_i^T)^{-1}, \\ \hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + K_k[\tilde{\mathbf{z}}_k^i - \tilde{h}(\hat{\mathbf{x}}_i^-, \mathbf{0})], \\ P_k^+ &= P_k^- - K_k \tilde{H}_i P_i^- F^T. \end{aligned} \tag{44}$$

- The predictive equation at extrapolated phase

$$\hat{\mathbf{x}}_{k+n}^- = f_{k+n-1}(\hat{\mathbf{x}}_{k+n-1}^-, \tilde{\mathbf{u}}_{k+n-1}, \mathbf{0}). \tag{45}$$

5. SIMULATIONS AND EXPERIMENTS

5.1. Simulations

In order to evaluate the validity of the proposed control algorithm, simulations were conducted in MATLAB [14]. Parameters for simulations are extracted from the real NRS described in next section. The process noise is modeled as being proportional to the angular speed $\omega_L(k)$ and $\omega_R(k)$ of the left and right wheels, respectively. The variances equal to $\delta\omega_L^2$ and $\delta\omega_R^2$, where δ is a constant determined by experiments as follows: the deviations between the true robot position and the position estimated by the kinematic model when driving the robot straight forwards several times (from the minimum to the maximum tangential speed of the robot) is determined. The deviations between the true robot orientation and the orientation estimated by the kinematic model when rotating the robot around its own axis several times (from the minimum to the maximum angular speed of the robot) is also determined. Based on those errors in position and orientation, the parameter δ is calculated. In our system, the δ is estimated to be 0.01. The input-noise covariance matrix Q_k is then given as

$$Q_k = \begin{bmatrix} \delta\omega_R^2(k) & 0 \\ 0 & \delta\omega_L^2(k) \end{bmatrix}. \tag{46}$$

The measurement is accomplished by two kinds of sensor: the optical encoder for the position (x, y) and the compass sensor for the orientation θ. The determination of the measurement-noise covariance matrix R_k is carried out similarly to the determination of the constant δ in which the measurement values are compared with the true values under different configurations in many times to estimate the expectation and variance of the

noise. In our system, the measurement-noise covariance matrix is identified as

$$R_k = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.018 \end{bmatrix}. \tag{47}$$

Remaining parameters are retrieved from the kinematic model of the robot (2) as follows

$$A_{k+1} = \left.\frac{\partial f_k}{\partial \mathbf{x}}\right|_{(\hat{\mathbf{x}}_k^+, \mathbf{u}_k, 0)} = \begin{bmatrix} 1 & 0 & -T_s v_k \sin\hat{\theta}_k^+ \\ 0 & 1 & T_s v_k \cos\hat{\theta}_k^+ \\ 0 & 0 & 1 \end{bmatrix}, \tag{48}$$

$$W_{k+1} = \left.\frac{\partial f_k}{\partial \mathbf{w}}\right|_{(\hat{\mathbf{x}}_k^+, \mathbf{u}_k, 0)} = T_s \frac{R}{2} \begin{bmatrix} \cos\hat{\theta}_k^+ & \cos\hat{\theta}_k^+ \\ \sin\hat{\theta}_k^+ & \sin\hat{\theta}_k^+ \\ \frac{2}{L} & \frac{2}{L} \end{bmatrix}, \tag{49}$$

$$H_k = V_k = I. \tag{50}$$

Parameters for the controller are chosen as follows: $\lambda = 6, h = 1, \gamma = 3$.

The first simulation evaluates the performance of the predictive filter PO-PF by comparing it with two popular localization algorithms including the extended Kalman filter (EKF) [9] and the optimal filter proposed in [15]. The comparison does not include the predictive phase as both the EKF and LEKF do not have this phase. The EKF is implemented with the assumption that it does not know if a measurement is delayed or not. It incorporates every measurement it receives as there is no delay. On the other band, the filter proposed in [15] is specifically designed for the system subject to random delay and packet drop. It is called Lucas-Extended Kalman filter (LEKF) in this study. The LEKF uses an infinite buffer to store and rearrange the delayed and out-of-order measurements. Loss measurements are stored as zero values. At each estimation step, the filter iteratively executes the Kalman equations from the initial to the being estimated state. This method was proven to be optimal.

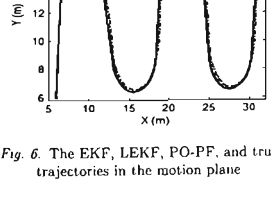


Fig. 6. The EKF, LEKF, PO-PF, and true trajectories in the motion plane

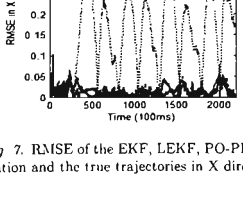


Fig. 7. RMSE of the EKF, LEKF, PO-PF estimation and the true trajectories in X direction

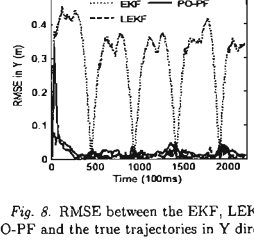


Fig. 8. RMSE between the EKF, LEKF, PO-PF and the true trajectories in Y direction

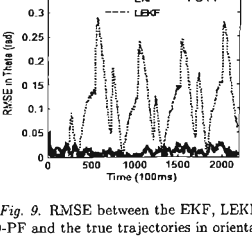


Fig. 9. RMSE between the EKF, LEKF, PO-PF and the true trajectories in orientation

In the simulation, the network parameters are set as an extreme scenario as follows: the time delay is between 800 ms and 1500 ms, the out-of-order rate is 15%, and the loss rate is 10%. Figure show the estimate and true trajectories in the motion plane. The overlap between the trajectories implies the good estimation. In order to realize the difference between algorithms, the root mean square error (RMSE) was calculated. Figs. 6-9 give the comparative curves in X, Y and θ directions simulated by 100-time Monte Carlo tests. We see that EKF has the largest error while the PO-PF and LEKF introduce equivalent small error. In addition to the accuracy, the computational burden is also considered. Standard MATLAB functions such as *tic toc* and *flops* [16] were used to calculate the amount of floating point operations and execution time of the filters. Tab. 1 shows the result scaled with respect to the EKF. It shows that the PO-PF is around two times higher than the EKF, but many times smaller than the LEKF.

Table 1. Normalized computational burden of filters

Parameter	EKF	LEKF	PO-EKF
Floating point operations	1.0	36.5	4.7
Execution time	1.0	33.7	2.4

The next simulation verifies the asymptotic stabilization of the control laws (11) for non-networked robot system. Figs. 10-11 present the results with the initial pose (-4, -3, 90^0) and the final pose (0, 0, 0^0). The (x, y) coordinate and θ orientation go toward the final state (0, 0, 0^0) while the tangent and angular velocities also go toward zeros. Thus, the system is asymptotically stable.

The third simulation applies the same stable control laws to the NRS. Simulation results with 500 ms network delay, 5% out-of-order, and 1% message loss are shown in Figs. 12-13. The robot's pose (x, y, θ) cannot reach (0, 0, 0^0) and the angular speed does not go to zero as expected. The system is therefore not asymptotically stable.

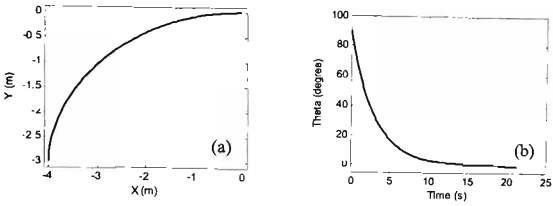


Fig. 10. Stabilization control of the non-networked robot: (a) Trajectory of the robot in the motion plane; (b) Variation of the direction of the robot

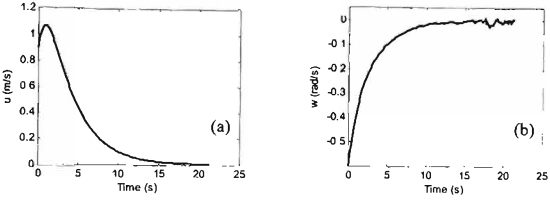


Fig. 11. Robot's velocities during the stabilization control of non-networked robot system: (a) Variation of the tangent velocity; (b) Variation of the angular velocity

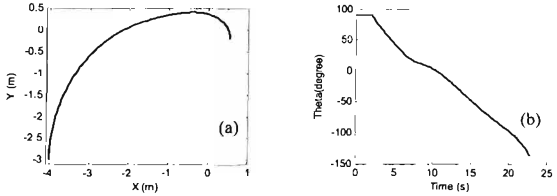


Fig. 12. Control of the NRS without using the PO-PF: (a) Trajectory of the robot in the motion plane; (b) Variation of the direction of the robot

Figs. 14-15 present the control results with the use of the PO-PF. It is able to see that the robot's pose and velocities go to zero again. The system is asymptotically stable. It confirms the success of the predictive filter in ensuring the stability of the NRS.

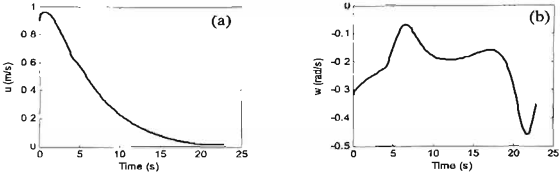


Fig. 13. Robot's velocities during the control of NRS without using the PO-PF: (a) Variation of the tangent velocity; (b) Variation of the angular velocity

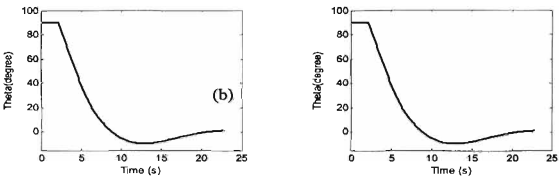


Fig. 14. Control of NRS with the use of the PO-PF: (a) Trajectory of the robot in the motion plane; (b) Variation of the direction of the robot

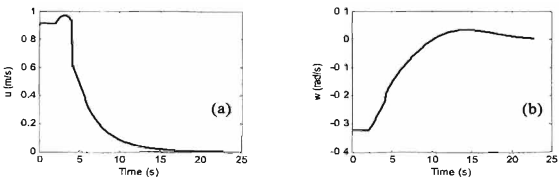


Fig. 15. Robot's velocities during the control of NRS with the use of the PO-PF: (a) Tangent velocity; (b) Angular velocity

5.2. Experiments

Experiments were conducted in a real NRS. The robot is a Multi-Sensor Smart Robot (MSSR) developed at our laboratory. It contains basic components for motion control, sensing, navigation (Figs. 16-17). The communication environment is the Internet. The remote controller is written in Visual C++ and implemented in a laptop computer. More details of the system can be referred from our previous work [17].

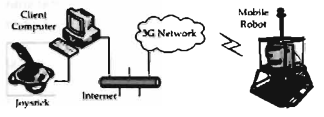
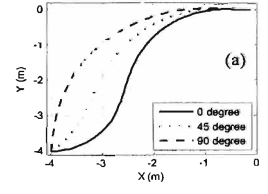


Fig. 16. Communication configuration of the networked robot system

Fig. 17. The hardware configuration of the robot



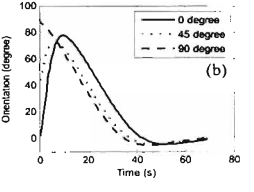
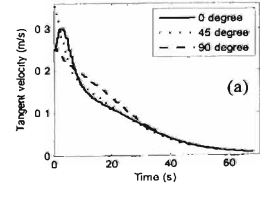


Fig. 18. Stable control of NRS with three different initial poses $(-4, -4, 0^0)$, $(-4, -4, 45^0)$, and $(-4, -4, 90^0)$: (a) Trajectories of the robot in the motion plane; (b) Variation of the direction of the robot



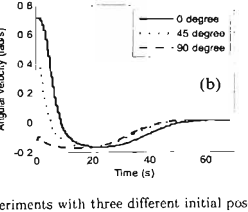


Fig. 19. Robot's velocities during stable control experiments with three different initial poses (-4,-4,00), (-4,-4,450), and (-4,-4,900): (a) Tangent velocity; (b) Angular velocity

In experiments, the goal is to navigate the mobile robot from starting points $(-4,-4,0),(-4,-4,45)$, and $(-4,-4,90)$ to reach the destination $(0,0,0^0)$. The parameters of the controller were chosen as: $\lambda = 200, h = 3, \gamma = 100$. The network parameters were measured as follows: the time delay was between 300 ms and 500 ms; the out-of-order rate was 2.4%; and the loss rate was 1.3%.

Fig. 18 describes the trajectories of the robot. Fig. 19 shows the tangent and angular velocities of the robot during the operation. They proved the success of robot in reaching the destinations and the convergence to zero of velocities at those positions. In another word, the system is asymptotically stable.

5.3. Discussion

We have carried out many experiments with different initial positions, network configurations, and control parameters (λ, h, γ). At all time, the robot succeeded in reaching the destinations. However, we also recognize that the control parameters (λ, h, γ) play an important role in determining the way the robot reaches the destination. For example, the high values of λ and γ speed up the goal reaching process but introduce more fluctuation to the trajectory while the high value of h implies faster change in robot's direction. Fig. 20 compares trajectories of the robot during the stable control from point $(-4,-4,0)$ to point $(0,0,0)$ with different control parameters. It is recognizable that the configuration with $\lambda = 200, h = 3, \gamma = 100$ gives the best result in term of traveling distance and time response. It suggests that tuning these parameters need be carefully taken into account when implementing practical systems as they can significantly enhance the control performance.

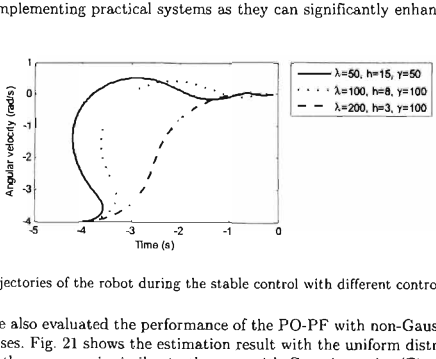


Fig. 20. Trajectories of the robot during the stable control with different control parameters

We have also evaluated the performance of the PO-PF with non-Gaussian and non-zero mean noises. Fig. 21 shows the estimation result with the uniform distribution noise. It shows that the accuracy is similar to the case with Gaussian noise (Figs. 6-9). Fig. 22 shows the estimation result with the noise having the mean of 20 cm. In this case, the accuracy is reduced comparing to the estimate with zero-mean noise (Figs. 6-9). It is concluded that the PO-PF still works with different types of noise but the accuracy may be slightly affected.

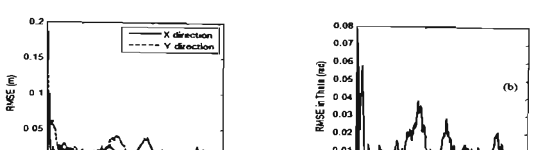


Fig. 21. Estimate by the PO-PF with uniform distribution noise: (a) RMSE in X and Y directions; (b) RMSE in orientation

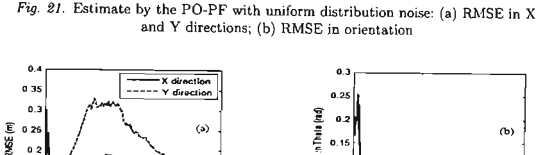


Fig. 22. Estimate by the PO-PF with non-zero mean noise: (a) RMSE in X and Y directions; (b) RMSE in orientation

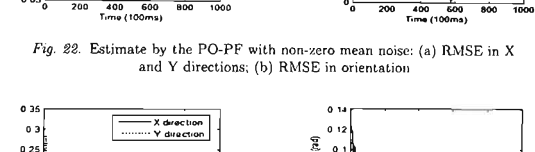


Fig. 23. RMSE of the predictive filter with the time-out value set to 5 seconds: (a) RMSE in X and Y directions, (b) RMSE in orientation

The next discussion is the acceptable error of the system at the destination position. More specifically, we stated from the problem formulation that the goal of the controller is to drive the error e between the present and final poses toward zero as the time goes infinite: $\lim_{t\to\infty} \mathbf{e}(t) = 0$. It means that it may take a very long time for the robot to exactly

reach the destination. In practice, this condition can be relaxed as each control task often tolerates certain error. For example, the time for our robot to reach the destination $(0, 0, 0)$ from point $(-4, -4, 0)$ with the error of 1 cm is 75 seconds. If the acceptable error is 5 cm, the goal reaching time reduces to 15 seconds. In conclusion, there is a trade off between the time response and the accuracy depending on control requirements.

Finally, it is noted that if the estimation is 100% accurate, the robot's pose will converge to zero as proven in section 3. In practice, error in estimate is unavoidable. If the error is sufficiently small, it can be considered as the measurement noise and the system is still asymptotically stable. However, as the time delay grows, the error is accumulated and in the worst case it can break the system stability. To avoid this, a time-out value is set. It is the maximum delay at which the estimate error still meets the system requirement. The requirement comes from the acceptable error discussed above. In our system, the time-out value is set to 5 seconds. This value ensures that the estimation error is within 30 cm which is the range required for the functioning operation of the system (Fig. 23). It is also noted that a delay longer than 5 seconds over the Internet would indicate a network congestion or interruption which stops the NRS from functioning operation.

6. CONCLUSION

This paper addresses the stabilization control problem of NRS subject to communication delay, loss, and out-of-order. The main contribution is the derivation of a new state estimation filter namely past observation-based predictive filter (PO-PF). This filter enables the prediction of system state from past measurements. When combined with stable control laws of non-networked robot, it ensures the asymptotic stability of the NRS. Simulations and experiments have been conducted. The results not only prove the validity of the proposed approach but also reveal some characteristics that we have discussed to improve.

REFERENCES

[1] M. Aicardi, G. Casalino, A. Bicchi, and A. Balestrino. Closed loop steering of unicycle like vehicles via Lyapunov techniques. *Robotics & Automation Magazine, IEEE*, **2**, (1), (1995), pp. 27–35.

[2] B. M. Kim and P. Tsiotras. Controllers for unicycle-type wheeled robots: Theoretical results and experimental validation. *IEEE Transactions on Robotics and Automation*, **18**, (3), (2002), pp. 294–307.

[3] C. Y. Chen, T. H. S. Li, Y. C. Yeh, and C. C. Chang. Design and implementation of an adaptive sliding-mode dynamic controller for wheeled mobile robots. *Mechatronics*, **19**, (2), (2009), pp. 156–166.

[4] M. Wargui, A. Tayebi, M. Tadjine, and A. Rachid. On the stability of an autonomous mobile robot subject to network induced delay In *Proceedings of the IEEE International Conference on Control Applications*, (1997), pp. 28–30.

[5] V. S. K. Chaitanya, P. D Patro, and P. K. Sarkar. Delay dependent stability in the real time control of a mobile robot using neural networks. In *the 2007 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, (2007), pp. 95–100.

[6] R. Luck and A. Ray. An observer-based compensator for distributed delays. *Automatica*, **26**, (5), (1990), pp. 903–908.

[7] N. Xi and T. J. Tarn. Stability analysis of non-time referenced internet-based telerobotic systems. *Robotics and Autonomous Systems*, **32**, (2), (2000), pp. 173–178.

[8] M. Wang and J. N. K. Liu. Interactive control for internet-based mobile robot teleoperation. *Robotics and Autonomous Systems*, **52**, (2), (2005), pp. 160–179.

[9] G. Bishop and G. Welch. An introduction to the Kalman filter. *Proc of SIGGRAPH, Course*, (2001).

[10] L. Teslić, I. Skrjanc, and G. Klancar. EKF-based localization of a wheeled mobile robot in structured environments. *Journal of Intelligent & Robotic Systems*, **62**, (2), (2011), pp. 187–203.

[11] S. A. Berrabah, H. Sahli, and Y. Baudoin. Visual-based simultaneous localization and mapping and global positioning system correction for geo-localization of a mobile robot. *Measurement Science and Technology*, **22**, (12), (2011), pp. 1–9.

[12] D. Simon. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, (2006).

[13] R. W. Brockett. *Asymptotic stability and feedback stabilization*. Defense Technical Information Center, (1983).

[14] A. Gilat. *MATLAB: An introduction with applications*. John Wiley & Sons, (2009).

[15] L. Schenato. Optimal estimation in networked control systems subject to random delay and packet drop. *IEEE Transactions on Automatic Control*, **53**, (5), (2008), pp. 1311–1317.

[16] T. T. Hoang, P M. Duong, N. T. T. Van, and T. Q. Vinh. Stabilization control of the differential mobile robot using Lyapunov function and extended Kalman filter. *Journal of Science and Technology, VAST*, **50**, (4), (2012), pp. 441–452

[17] P. M. Duong, T T. Hoang, N. T. T. Van, D. A. Viet, and T. Q. Vinh. A novel platform for internet-based mobile robot systems. In *the 7th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, (2012), pp. 1972–1977.