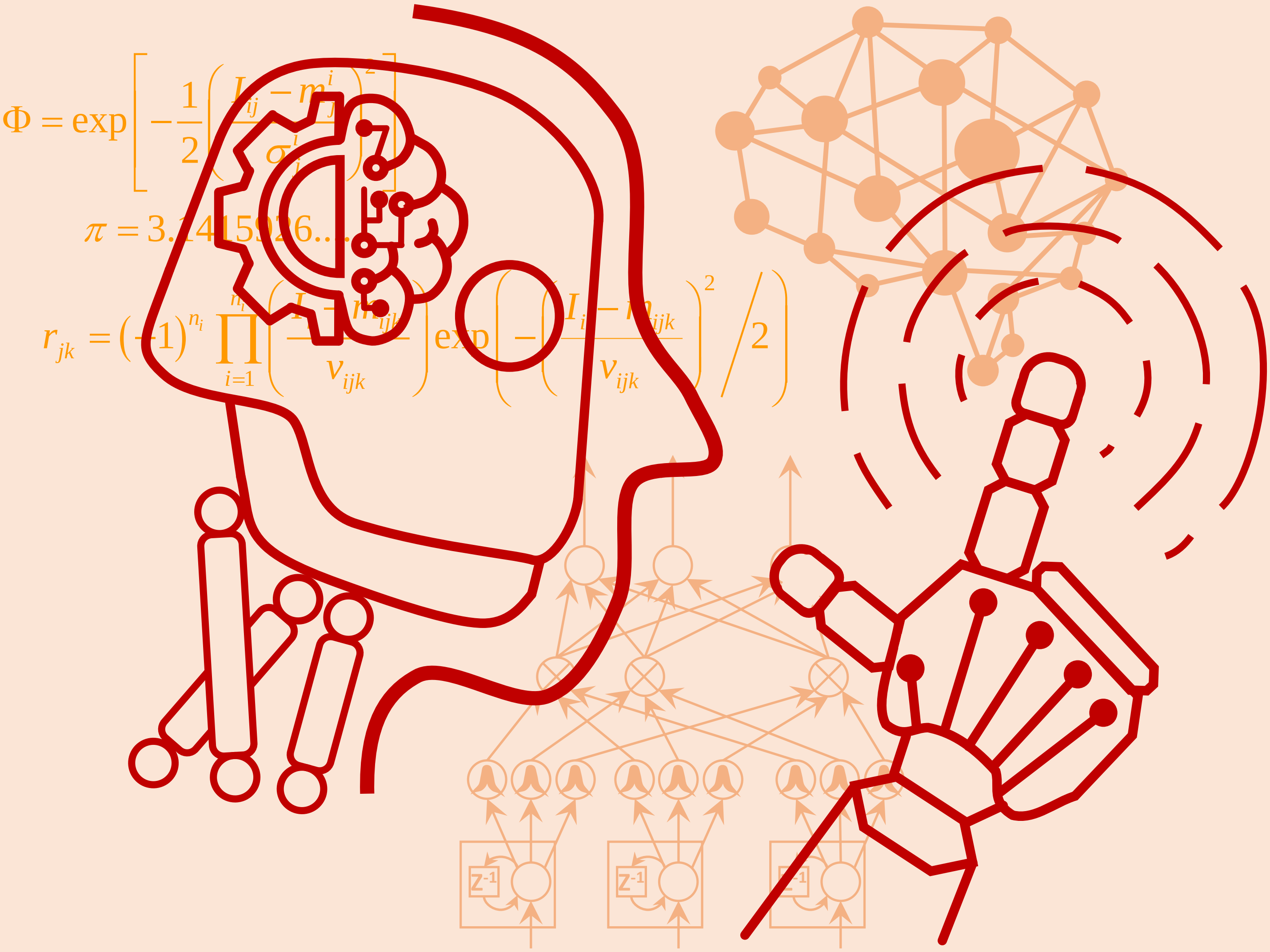
ISSN 2616-8170

RST
Robotics Society of Taiwan

iRobotics

VOLUME 2
NUMBER 1
MARCH 2019


$\Phi = \exp\left[-\frac{1}{2}\left(\frac{I_{ij}-m_{j}^{i}}{\sigma_{j}^{i}}\right)^{2}\right]$
$\pi = 3.1415926...$
$r_{jk} = (-1)^{n_i}\prod_{i=1}^{n_i}\left(\frac{I_i - m_{ijk}}{v_{ijk}}\right)\exp\left(-\left(\frac{I_i - m_{ijk}}{v_{ijk}}\right)^{2}/2\right)$
z^{-1}
z^{-1}
z^{-1}

PUBLISHED BY THE ROBOTICS SOCIETY OF TAIWAN

*i*Robotics

EDITORIAL BOARD

PUBLISHER

Robotics Society of TAIWAN (RST)
Society President: Ching-Chih Tsai

Department of Electrical Engineering, National Chung Hsing University
Taichung, Taiwan

Tel: +886-4-2285-1549#601
URL: http://www.rst.org.tw

The ***i*Robotics** is published quarterly each year by the Robotics Society of Taiwan (RST). Institutional rate: US$140 annually; individual annual subscription rate: US$50 for nonmembers, US$25 for members (including postage). Note that another US$100 is needed if the express is required.

Intelligent Adaptive Simultaneous Tracking and Stabilization Using Fuzzy Wavelet Networks for a Wheeled Inverted Pendulum

Ching-Chih Tsai, *Fellow, IEEE*, Chien-Cheng Yu, Shih-Min Hsieh and Feng-Chun Tai

***Abstract*- This paper presents a backstepping sliding-mode motion control method for simultaneous tracking and stabilization of a wheeled inverted pendulum. Based on the nonlinear modeling of the vehicle incorporating static friction between the wheels and the motion surface, a decoupling approach is utilized to decompose the overall system into two subsystems: an inverted pendulum subsystem and a yaw motion subsystem, and the backstepping technique together with the sliding mode control method is then used to synthesize a unified controller in both kinematic and dynamic level. In the kinematic level, a virtual control is designed to accomplish simultaneous tracking and stabilization, while in the dynamic level, two sliding-mode controllers are designed such that the robot can be steered to follow desired trajectories or go from one pose to another pose. Simulations results and experimental results are conducted to show that the proposed unified control method is capable of satisfactorily achieving both tracking and regulation.**

Index Terms—backstepping, stabilization, sliding-mode control, trajectory tracking, wheeled inverted pendulum.

I. Introduction

RECENTLY, wheeled inverted pendulums have been widely investigated in both academia and industry; and they have been successfully applied to construct several autonomous service robots, such as soccer robots, reception robots. The wheeled inverted pendulums are very useful in constructing many kinds of service robots working with people. Hosoda et al. [1] detailed the basic design of human-symbiotic robot EMIEW whose linear motion speed was up to 1.67 m/sec; their EMIEW was designed using a two-processor system structure. On the other hand, many researchers [1]-[15] have shown that the wheeled inverted pendulums have gained many applications, including self-balancing two wheeled transportation and soccer games. With the advent of modern control technology, such platforms with sophisticated control functions and safety features can be cost down so that they have highly potential to satisfy stringent requirements of various autonomous service robots with high linear speed.

Design, modeling and control of wheeled inverted pendulums have been widely studied by several researchers. Lin [2] constructed a nonlinear mathematical modeling for a kind of wheeled inverted pendulum considered as two-wheeled self-balancing mobile platform. Sasaki et al. [3] constructed a lightweight self-balancing personal riding-type wheeled mobile platform (PMP); the PMP steering control was achieved by changing the position of the rider’s center of gravity. Grasser et al. [4] presented an unmanned mobile inverted pendulum, and Pathak et al. [5] studied the dynamic equations of the wheeled inverted pendulum by partial feedback linearization. Ha et al. [7] presented the trajectory tracking system for navigation of the inverse pendulum type self-contained mobile robot; however, this method was limited to simple straight line motion and simple turning. For both trajectory tracking and stabilization problems, much effort has been spent on nonholonomic mobile robots in [16, 17], but little attention has been paid to address the same problems for wheeled inverted pendulums. Furthermore, no attempt has been paid to design a simultaneous tracking and stabilization controller for wheeled inverted pendulums with frictions and uncertainties caused by different payloads and terrain [1, 2, 5, 7].

From controller design of view, the control of the wheeled inverted pendulums can be thought of as an under-actuated control problem, which has been investigated by several researchers [18-20]. In particular, Lo and Kuo [18] provided a decoupled sliding-mode control to stabilize a nonlinear system with four state variables, Lin and Mon [19] offered a hierarchical decoupling sliding-mode control to regulate a more general class of under-actuated control systems, and Wang et al. [20] presented two systematic sliding-mode design methods for a class of under-actuated mechanical systems. However, the approaches [18-20] have not been applied to the wheeled inverted pendulums yet!

Fuzzy wavelet networks (FWN) have been proved to excellently approximate time-varying nonlinear functions or nonlinear dynamics [21]. This property can be easily applied to controller design. For example, Lin [22] brilliantly used FWN to on-line learn a nonsingular terminal sliding mode controllers for robot manipulators, thus accomplishing out excellent trajectory tracking performance.

Hence, the goals of this paper are to apply backstepping technique and sliding mode control approach to develop a simultaneous tracking and stabilization (STS) controller, and to verify it by using simulations and experimentations on an experimental wheeled inverted pendulum. The proposed control method will be shown to be useful and powerful in achieving satisfactory trajectory tracking and stabilization in presence of parameter variations, uncertainties and frictions.

Ching-Chih Tsai, Chien-Cheng Yu, Shih-Min Hsieh and Feng-Chun Tai are with the Department of Electrical Engineering, National Chung Hsing University, Taichung, 40227, Taiwan.

Chien-Cheng Yu is also with the Department of Electronic Engineering, Hsiuping University of Science and Technology, Taichung, Taiwan.

(Corresponding author Ching-Chih Tsai, email: cctsai@nchu.edu.tw) (email:ccyu@hust.edu.tw,shiesh2000@mail.post.gov.tw,fctai@nchu.edu.tw)

The authors deeply acknowledge final support in part from National Science Council, Taiwan, ROC, under contract 96-2221-E-005-106-MY2, and in part from the ministry of education, Taiwan, ROC, under ATU plan.

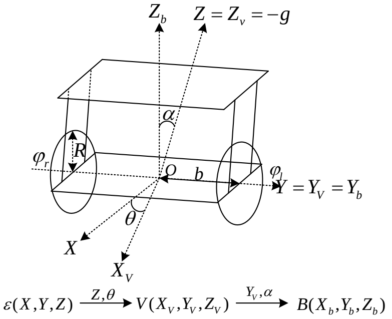

Z_b
$Z = Z_v = -g$
α
R
φ_r
O
b
φ_l
$Y = Y_V = Y_b$
θ
X
X_V
$\varepsilon(X,Y,Z) \xrightarrow{Z,\theta} V(X_V,Y_V,Z_V) \xrightarrow{Y_V,\alpha} B(X_b,Y_b,Z_b)$

Fig. 1. Schematic diagram of the wheeled inverted pendulum [5].

The unified STS controller not only is of theoretical interest, but also has some practical advantages over the use of separate stabilization and tracking controllers, such as only one controller and transient improvement because of no switching. This paper is written in two principal contributions; one is that the proposed STS controller is proven globally asymptotically stable via the Lyapunov stability theory, and the other is that the merit and applicability of the proposed controller is exemplified by means of an experimental wheeled inverted pendulum.

The rest of the paper is outlined as follows. Section II revisits the mathematical modeling of the wheeled inverted pendulum and briefly states the problem formulation. Section III presents the STS controller for the wheeled inverted pendulum. In Section IV, simulations are several experiments are conducted to show the effectiveness and performance of the proposed control method. Section V concludes the paper.

II. Dynamic Model and Problem Statement

A. Mathematical Modeling

To steer the wheeled inverted pendulum, mathematical model is a necessary such that the robot with the designed controllers can successfully achieve desired control objectives. Fig. 1 depicts the free body diagram of the mobile inverted pendulum. Since the procedure to establish a mathematical model of the mobile inverted pendulum has been derived in some detail in [5], the section will omit the similar modeling process, but put emphasis on the difference. The presented model not only has the same equations of motion of the vehicle, but also includes viscous friction, which depends on the moving speed of the platform, and static friction opposite to the moving direction. For the sake of simplicity, Table 1 lists all the used symbols and their definitions. By using the Lagrange's mechanics [5], it is easy to construct the nonlinear system model of the vehicle with both viscous and static frictions in the following state-space form.

$$\mathbf{M(q)\ddot{q} + V(q,\dot{q}) = E(q)\tau + A^T(q)}\lambda \tag{1}$$

where $\mathbf{q}_{6\times 1} = [x \quad y \quad \theta \quad \alpha \quad \phi_r \quad \phi_l]$ represents the configuration variables of the system; the pair (x, y) denoting the position of the vehicle, θ the orientation, α the tilt angle, and both ϕ_r and ϕ_l the encoder information on the right and left wheels, respectively. Moreover, λ is the constraint-force vector, and $\tau = \begin{bmatrix}\tau_r & \tau_l\end{bmatrix}^T$ is the torque vector. In (1),

Table 1. List of the parameters and variables

Symbol [unit]	Parameter and variable name
I_{xx}, I_{yy}, I_{zz}	Moment of inertia of the pendulum body
V_r [m/sec]	Reference linear velocity
ω_r [m/sec]	Reference angular velocity
R [m]	Radius of the wheels
c_x, c_z	The center of mass of the pendulum body G_b
τ	Input torque vector for right and left motors
τ_r	Input torque applied to the right motor
T_l	Input torque applied to the left motor
I_{wa}, I_{wd} [$Kg\ m^2$]	Moment of inertia of a wheel about its axis and diameter
ϕ_r, ϕ_l [rad]	Angles of the right and left wheels
θ [rad]	Yaw angle
M_b [Kg]	Mass of the pendulum body
M_w [Kg]	Mass of the each wheel
α [rad]	Tilt angle of the wheeled inverted pendulum
b [m]	Half of the distance between both driving wheels
(x, y)[m/sec]	Position of the wheeled inverted pendulum
v, ω	Linear and angular velocities of the wheeled inverted pendulum

$$\boldsymbol{E(q)} = \begin{bmatrix} 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}^T$$

$$\boldsymbol{A(q)}_{3\times 6} = \begin{bmatrix} -\sin(\theta) & \cos(\theta) & 0 & 0 & 0 & 0 \\ \cos(\theta) & \sin(\theta) & b & 0 & -R & 0 \\ \cos(\theta) & \sin(\theta) & -b & 0 & 0 & -R \end{bmatrix} \tag{2}$$

Note that the three nonholonomic constraint of the vehicle due to no-slip is expressed by $\mathbf{A(q)}_{\mathbf{3?6}}\dot{\mathbf{q}} = \mathbf{0}$, and the null-space of $A(q)$ is given by the matrix $\mathbf{S(q)}$ as

$$\mathbf{S(q)}_{6\times 3} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ \cos(\theta) & \sin(\theta) & 0 & 0 & 1/R & 1/R \\ 0 & 0 & 1 & 0 & b/R & -b/R \end{bmatrix}^T \tag{3}$$

With the matrix $S(q)$, one obtains $\dot{q}(t) = S(q)v(t)$. Further, the Lagrange multipliers λ in (1) can be eliminated by pre-multiplying with S^T , thus obtaining

$$(S^T MS)\dot{v} + S^T(M\dot{S}v + V(q,\dot{q})) = S^T E(q)\tau \tag{4}$$

To have a more useful system model, one discards the last two variables of the configuration variables and re-define the following two vectors, q_r and V_r, and the augmented state vector **x**.

$$q_r \triangleq \begin{bmatrix} x \\ y \\ \theta \\ \alpha \end{bmatrix},\ V_r = \begin{bmatrix} \dot{\alpha} \\ v \\ \dot{\theta} \end{bmatrix} \triangleq \begin{bmatrix} \omega_\alpha \\ v \\ \omega \end{bmatrix},\ \mathbf{x} = \begin{bmatrix} q_r \\ V_r \end{bmatrix} \tag{5}$$

Taking the time derivatives of the augmented state vector **x** gives

$$\dot{\mathbf{x}} = \mathbf{f(x)} + \mathbf{g(x)}\tau \tag{6}$$

where $\mathbf{f(x)} = \begin{bmatrix} \mathbf{f_1(x)} \\ \mathbf{f_2(x)} \end{bmatrix}$, $\mathbf{g(x)} = \begin{bmatrix} \mathbf{g_1(x)} \\ \mathbf{g_2(x)} \end{bmatrix}$ and

$$f_1(\boldsymbol{x}) = \left[\cos(\theta)v \quad \sin(\theta)v \quad \omega \quad \omega_\alpha\right]^T \ , \quad g_1(\boldsymbol{x}) = [\boldsymbol{0}]_{4\times 2}$$

$$f_2(\boldsymbol{x}) \triangleq \left[f_{21}(\boldsymbol{x}) \quad f_{22}(\boldsymbol{x}) \quad f_{23}(\boldsymbol{x})\right]^T$$

$$= -\left(S^T MS\right)^{-1} S^T \left(M\dot{S}v + V(q,\dot{q})\right) \begin{bmatrix} \frac{1}{D_\alpha}(\sin(2\alpha)\dot{\theta}^2\overline{H}) + \frac{1}{2D_\alpha}(M_b{}^2c_z{}^2R^2\sin(2\alpha)(\dot{\alpha})^2) + \frac{1}{2D_\alpha}(-2M_b{}^2R^2c_z - 4I_{wa}M_bc_z - 4M_wR^2M_bc_z)g\sin(\alpha) \\ K_\alpha\dot{\theta}^2 + \frac{1}{2D_\alpha}(M_b{}^2c_z{}^2R^2g\sin(2\alpha)) + \frac{1}{4D_\alpha}(-4I_{yy}M_bR^2c_z - 4R^2M_b{}^2c_z{}^3)\sin(\alpha)(\dot{\alpha})^2 \\ \frac{1}{G_\alpha}(-(I_{xx} - I_{yy})R^2 - M_bc_z{}^2R^2)\sin(2\alpha)\dot{\alpha}\dot{\theta} - \frac{1}{G_\alpha}(\sin(\alpha)R^2M_bc_zv\dot{\theta}) \end{bmatrix}$$

$$g_2(\boldsymbol{x}) = (S^TMS)^{-1}S^TE(q) \triangleq \begin{bmatrix} g_{21}(\boldsymbol{x}) & g_{21}(\boldsymbol{x}) \\ g_{22}(\boldsymbol{x}) & g_{22}(\boldsymbol{x}) \\ g_{23}(\boldsymbol{x}) & -g_{23}(\boldsymbol{x}) \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{D_\alpha}(M_bR^2 + 2M_wR^2 + 2I_{wa} + M_b\cos(\alpha)c_zR) & \frac{1}{D_\alpha}(M_bR^2 + 2M_wR^2 + 2I_{wa} + M_b\cos(\alpha)c_zR) \\ -\frac{R}{D_\alpha}(M_b\cos(\alpha)c_zR + I_{yy} + M_bc_z{}^2) & -\frac{R}{D_\alpha}(M_b\cos(\alpha)c_zR + I_{yy} + M_bc_z{}^2) \\ \frac{R\cdot b}{G_\alpha} & -\frac{R\cdot b}{G_\alpha} \end{bmatrix}$$

$$D_\alpha = M_b{}^2\cos^2(\alpha)c_z{}^2R^2 + ((-M_b{}^2 - 2M_wM_b)c_z{}^2 - 2I_{yy}M_w - I_{yy}M_b)R^2 - 2M_bc_z{}^2I_{wa} - 2I_{yy}I_{wa}$$

$$G_\alpha = (-M_bc_z{}^2 + I_{zz} - I_{xx})R^2\cos^2(\alpha) \ + (M_bc_z{}^2 + I_{xx} + 2I_{wd} + 2b^2M_w)R^2 + 2b^2I_{wa}$$

$$K_\alpha(4D_\alpha) = (-4I_{yy}M_bR^2c_z - 3R^2M_b{}^2c_z{}^3 + M_bR^2c_z(I_{xx} - I_{yy}))\sin(\alpha) \ + (M_bR^2c_z(I_{xx} - I_{zz}) + R^2M_b{}^2c_z{}^3)\sin(3\alpha)$$

$$\overline{H} = \frac{1}{2}M_bR^2I_{zz} + I_{wa}I_{zz} - M_wR^2I_{xx} - I_{wa}I_{xx} - M_bc_z{}^2M_wR^2 - M_bc_z{}^2I_{wa} - \frac{1}{2}M_bR^2I_{xx} + M_wR^2I_{zz}$$

With the use of the transformation of both yaw and pitch torques, τ_y and τ_p, into the wheel torques τ_l and τ_r,

$$\begin{bmatrix} \tau_l \\ \tau_r \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & -0.5 \end{bmatrix} \begin{bmatrix} \tau_p \\ \tau_y \end{bmatrix} \tag{7}$$

and defining $\overline{\tau} = \left[\tau_p \quad \tau_y\right]^T$, (6) becomes

$$\dot{\mathbf{x}} = f(\mathbf{x}) + \begin{bmatrix} 0_{4\times 2} \\ \overline{g}_2(\mathbf{x}) \end{bmatrix} \overline{\tau} \tag{8}$$

where

$$\overline{g}_2(\mathbf{x}) \triangleq \begin{bmatrix} g_{21}(\mathbf{x}) & 0 \\ g_{22}(\mathbf{x}) & 0 \\ 0 & g_{23}(\mathbf{x}) \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{D_\alpha}(M_bR^2 + 2M_wR^2 + 2I_{wa} + M_b\cos(\alpha)c_zR) & 0 \\ -\frac{R}{D_\alpha}(M_b\cos(\alpha)c_zR + I_{yy} + M_bc_z{}^2) & 0 \\ 0 & \frac{R\cdot b}{G_\alpha} \end{bmatrix} \tag{9}$$

From (8), it follows that the dynamic model of the pendulum is governed by two parts of equations: kinematic and dynamic.

$$\dot{q}_r = \left[\dot{x} \quad \dot{y} \quad \dot{\theta} \quad \dot{\alpha}\right]^T = \left[v\cos\theta \quad v\sin\theta \quad \omega \quad \omega_\alpha\right]^T \tag{10}$$

and

$$\dot{V}_r = \begin{bmatrix} \dot{\omega}_\alpha \\ \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} f_{21}(x) \\ f_{22}(x) \\ f_{23}(x) \end{bmatrix} + \begin{bmatrix} g_{21}(x) & 0 \\ g_{22}(x) & 0 \\ 0 & g_{23}(x) \end{bmatrix} \begin{bmatrix} \tau_p \\ \tau_y \end{bmatrix} \tag{11}$$

The kinematic part reveals the relations between the position, orientation and inclination of the pendulum and their velocities, whereas the dynamic part involves with the relations between the three accelerations and the two torques, τ_p and τ_y. Furthermore, from (10) and (11), it indicates that two controllers for τ_p and τ_y can be synthesized independently from each other and then combined together to accomplish the control goal.

B. Problem Formulation

The design goal of the STS control method for the pendulum system model described by (10) and (11) is to keep the trajectories of the pendulum asymptotically follow time-varying reference trajectories or desired positions and orientations of a fixed reference configuration in one unified control framework, and to main the tilt angle of the pendulum at origin. To formulate the problem, let $\tilde{x}(t)$, $\tilde{y}(t)$, $\tilde{\theta}(t)$ be the differences between the real position $x_c(t)$, $y_c(t)$ and the angle $\theta(t)$ of the nonholonomic mobile robot with the desired reference trajectory, $q_{rc}(t)=[x_{rc}(t),\ y_{rc}(t),\ \theta_r(t)]^T\in R^3$, in the Cartesian coordination, i.e.,

$$\tilde{x}(t) = x_c - x_{rc} \ , \ \tilde{y}(t) = y_c - y_{rc} \ , \ \tilde{\theta}(t) = \theta_c - \theta_{rc} \tag{12}$$

Moreover, the desired reference trajectory $q_{rc}(t)$ satisfies the following kinematics equation.

$$\dot{x}_{rc}(t) = v_r(t)\cos(\theta_r)\ ,\ \ \dot{y}_{rc}(t) = v_r(t)\sin(\theta_r)\ ,\ \ \dot{\theta}_r = \omega_r(t) \quad (13)$$

where $v_r(t)$ and $\omega_r(t)$ represents the desired time varying linear and angular velocities.

The aim of the STS control method is to design the two torques, τ_p and τ_y, such that $\tilde{x}(t) \to 0$, $\tilde{y}(t) \to 0$, $\tilde{\theta}(t) \to 0$ as $t \to \infty$ and the pitch angle α is maintained at zero. Worthy of mention is that, in (13), if $v_r(t)$ and $\omega_r(t)$ are zero, then the desired reference trajectory $q_{rc}(t)$ becomes a fixed reference configuration, and the simultaneous and stabilization law turns out a regulation law.

III. Controller Synthesis

This section is devoted to applying the backstepping technique to develop a STS controller for wheeled inverted pendulum. The design procedure is divided into two levels: kinematic and dynamic. In the kinematic level, virtual controls for $v(t)$, $\omega_r(t)$ and $\omega_\alpha(t)$ are designed based on the subsystem (10), in order to stabilize the tilt angle and achieve that $\tilde{x}(t) \to 0, \tilde{y}(t) \to 0, \tilde{\theta}(t) \to 0$ as $t \to \infty$, while; in the dynamic level, the control laws for the two torques, τ_p and τ_y, are established based on the subsystem (11) and three backstepping errors, which are the difference between the virtual controls and actual trajectories. The following two subsections elaborate the controller synthesis in both levels.

A. Kinematic Level : Virtual Motion Command Generator

For the subsystem (10) describing kinematic behavior of the vehicle pose and its inclination, the three variables $v(t)$, $\omega_r(t)$ and $\omega_\alpha(t)$ are regarded as virtual controls. To stabilize the inclination of the vehicle, it is easy to propose the virtual control

$$\omega_\alpha(t) \triangleq \phi_1(\alpha) = -k_\alpha \alpha \quad (14)$$

so as to regulate the tilt angle to zero exponentially. On the other hand, to achieve $\tilde{x}(t) \to 0, \tilde{y}(t) \to 0, \tilde{\theta}(t) \to 0$ as $t \to \infty$ in the kinematic level, one considers the first three equations of the subsystem (10) as a well-known kinematic model of a mobile robot with differential driving, and apply the STS control approach developed by Dixon et al. [23] to the STS goal. The following elucidates some details of our STS controller, which will be later shown to be superior to the one by Dixon et al. [23], because the STS controller adopts the following new globally invertible transformation.

$$\begin{bmatrix} w \\ z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} -k\tilde{\theta}\cos\theta + 2k\sin\theta & -k\tilde{\theta}\sin\theta - 2k\cos\theta & 0 \\ \cos\theta & \sin\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{\theta} \end{bmatrix} \quad (15)$$

where $k>0$, $w(t) \in R^1$ and $z(t) = [z_1(t)\ z_2(t)]^T \in R^2$ are the auxiliary tracking error vectors, and $\tilde{x}(t),\ \tilde{y}(t),\ \tilde{\theta}(t) \in R^1$ are given in (12). From (6), we have the inverse transformation

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{\theta} \end{bmatrix} = \begin{bmatrix} \frac{1}{2k}\sin\theta & \frac{1}{2k}\left(z_2\sin\theta + 2k\cos\theta\right) & 1 \\ -\frac{1}{2k}\cos\theta & -\frac{1}{2k}\left(z_2\cos\theta + \sin\theta\right) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} w \\ z_1 \\ z_2 \end{bmatrix} \quad (16)$$

Note that, from (16), if $w(t)$, $z_1(t)$, $z_2(t) \in L_\infty$, then $\tilde{x}(t),\ \tilde{y}(t),\ \tilde{\theta}(t) \in L_\infty$. Moreover, it is easy to show that $\lim_{t\to\infty}\left(w(t), z_1(t), z_2(t)\right) = 0$, then

$$\lim_{t\to\infty}\left(w(t), z_1(t), z_2(t)\right) = 0 \quad (17)$$

Furthermore, differentiating (15) and using (10),(12-13) obtain

$$\dot{w} = \left(\begin{bmatrix} k & k\left(-\tilde{x}\sin\theta + \tilde{y}\cos\theta\right) \\ 0 & k \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} - \begin{bmatrix} kv_r\cos z_2 \\ k\omega_r \end{bmatrix} \right)^T \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + 2k\left(z_1\omega_r - v_r\sin z_2\right) \quad (18)$$

$$\dot{z} = \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} 1 & -\left(\tilde{x}\sin\theta - \tilde{y}\cos\theta\right) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} - \begin{bmatrix} v_r\cos z_2 \\ \omega_r \end{bmatrix} \quad (19)$$

Combining (18) and (19) in a vector-matrix form, the dynamics of the tracking error becomes

$$\begin{aligned} \dot{w} &= k\mu^T J^T z + kf \\ \dot{z} &= \mu \end{aligned} \quad (20)$$

where $J^T \in R^{2\times2}$ is the skew-symmetry constant matrix defined by

$$J^T = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \quad (21)$$

and $f \in R^1$ is an auxiliary signal given by

$$f = 2\left(z_1\omega_r - v_r\sin z_2\right) \quad (22)$$

Further, $\mu(t) = [\mu_1(t)\ \mu_2(t)]^T \in R^2$ is called the auxiliary control input having the following form

$$\boldsymbol{\mu} = T^{-1}\begin{bmatrix} v \\ \omega \end{bmatrix} - \begin{bmatrix} v_r\cos z_2 \\ \omega_r \end{bmatrix} \quad (23)$$

where $T(t) \in R^{2\times2}$ is expressed by

$$\mathbf{T} = \begin{bmatrix} 1 & \left(\tilde{x}\sin\theta - \tilde{y}\cos\theta\right) \\ 0 & 1 \end{bmatrix} \quad (24)$$

To achieve STS control goal, one defines an auxiliary error signal $\tilde{z}(t) \in R^2$, where $\tilde{z}(t)$ is the difference between the auxiliary signal $z_d(t) \in R^2$ and the auxiliary tracking error vector $z(t)$, i.e.

$$\tilde{z}(t) = z_d - z \quad (25)$$

To stabilize the open loop error system of (20), one further defines another auxiliary control signal *u*(*t*) in (23) by

$$\mu = \mu_a - k_2 z \tag{26}$$

where the control term $\mu_a(t) \in R^2$ is given by

$$\mu_a = \left(\frac{k_1 w + f}{\delta_d^2}\right) J z_d + \Omega_1 z_d \tag{27}$$

Hence, the auxiliary signal $z_d(t)$ in (25) can be given by

$$\dot{z}_d = \left(\frac{k_1 w + f}{\delta_d^2} + kw\Omega_1\right) J z_d + \frac{\dot{\delta}_d}{\delta_d}\ \mathrm{z_d} \tag{28}$$

where $z_d^T z_d(0) = \delta_d^2(0)$, and the auxiliary control item $\Omega_1(t) \in R^1$ and $\delta_d(t) \in R^1$ are expressed by

$$\Omega_1 = k_2 + \frac{\dot{\delta}_d}{\delta_d} + kw\left(\frac{k_1 w + f}{\delta_d^2}\right) \tag{29}$$

$$\delta_d = \alpha_0 e^{-\alpha_1 t} \tag{30}$$

where $f(z.v_r.t)$ is defined in (22), and $k,\ k_1,\ k_2,\ \alpha_0,\ \alpha_1 \in R^1$ are positive constant control gains.

Remark 1: Based on (28) the definition of $\delta_d(t)$, equations (27-30) may have singularities, namely, as $\delta_d(t)$ decays exponentially to zero, the terms in (27-30), such as $\left(k_1 w + k_3 \tilde{z}^T z + f\right) J z_d \delta_d^{-2}$, $w\left(k_1 w + k_3 \tilde{z}^T z + f\right) J z_d \delta_d^{-2}$, $w^2\left(k_1 w + k_3 \tilde{z}^T z + f\right) J z_d \delta_d^{-2}$ will be unbounded as *t* approaches ∞. In the next section we will show how the singularity can be circumvented if appropriate control gains are selected.

Remark 2: Differentiating $z^t_D(t) z_d(t)$ with respect to time yields

$$\frac{d}{dt}\left(z_D^t z_d\right) = 2 z_d^T \left[\left(\frac{k_1 w + k_3 \tilde{z}^T z + f}{\delta_d^2} + kw\Omega_1\right) J z_d + \frac{\dot{\delta}_d}{\delta_d} z_d\right] \tag{31}$$

which, with (28), leads to

$$\frac{d}{dt}\left(z_D^t z_d\right) = 2\frac{\dot{\delta}_d}{\delta_d} z_d^T z_d \tag{32}$$

Using (28), we have

$$z_d^T(t) z_d(t) = \left\| z_d(t) \right\|^2 = \delta_d^2 \tag{33}$$

which is the unique solution of (32), where ||•|| is standard Euclidean norm.

To develop the closed-loop error control system of (20), one substitutes (27) into (20) to obtain

$$\dot{w} = k\left(\mu_a - k_2 z\right)^T J^T z + kf = k\mu_a^T J^T z + kf \tag{34}$$

and then substitutes (21) into (34) to have

$$\dot{w} = k\mu_a^T J \tilde{z} + k\mu_a^T J^T z_d + kf \tag{35}$$

Moreover, by substituting (27) into (35) and using (33), the closed-loop error system for *w*(*t*) is obtained from

$$\dot{w} = k\mu_a^T J \tilde{z} - kk_1 w \tag{36}$$

Next, by differentiating $\tilde{z}(t)$ with respect to time, and substituting $u_a(t)$ in (27) and $\dot{z}$ in (20) into $\dot{\tilde{z}}(t)$ and utilizing (27-29), one obtains dynamics of the closed-loop error system for $\tilde{z}(t)$ governed by

$$\dot{\tilde{z}} = kwJ\mu_a - k_2 \tilde{z} \tag{37}$$

The globally exponentially stability of the aforementioned error system can be easily proven by selecting a radially, unbounded and quadratic Lyapunov function $V(t) \in R^1$ by

$$V(t) = \frac{1}{2} w^2 + \frac{1}{2} \tilde{z}^T \tilde{z} \tag{38}$$

This main result is summarized as below.

Theorem 1: Let k, k_1, k_2, α_1 be real positive constants satisfying $\min(kk_1,\ k_2) > \alpha_1$. Then the control laws (26-30) make the closed-loop error system (20) globally exponentially stable, i.e., $\left|\tilde{x}(t)\right|,\ \left|\tilde{y}(t)\right|,\ \left|\tilde{\theta}(t)\right| \le \alpha_2 e^{-\alpha_3 t}$, where $\alpha_2 \in R^1$ and $\alpha_3 \in R^1$ are positive constants.

Remark 3: The virtual controls for v and ω are obtained from (23) and expressed by

$$\begin{bmatrix} v \\ \omega \end{bmatrix} \triangleq \begin{bmatrix} \phi_2 \\ \phi_3 \end{bmatrix} = T\mu + T\begin{bmatrix} v_r \cos z_2 \\ \omega_r \end{bmatrix} \tag{39}$$

B. Dynamic Level: Sliding-Mode Torque Controller

In this subsection, the dynamic part (11) of the vehicle will be employed to design two control laws for the two torques, τ_p and τ_y, using the backstepping sliding-mode control method. Such a controller is synthesized via two control modules. One is called sliding-mode yaw rate torque control module that involves with how the torque τ_y is generated such that the actual yaw rate tracks the virtual command ϕ_3 in (14) ; the other is named by sliding-mode posture and speed torque control module that tracks both virtual commands ϕ_1 and ϕ_2 such that the wheeled inverted pendulum eventually follows a desired speed velocity command v_r and maintains the pendulum without falling in finite time. In the sequel, the design procedures of both control modules are elaborated as well.

1) Sliding-Mode Yaw Rate Torque Control Module

Since the virtual control for ω has been developed in (39) and the dynamic part is decoupled for the torque τ_y, the backstepping error for ω is then given by

$$\eta_\omega = \omega - \phi_3 \tag{40}$$

and the sliding-mode surface for ω is chosen by $S_\omega = \eta_\omega$. Differentiating for S_ω gives

$$\dot{S}_\omega = \dot{\omega} - \dot{\phi}_3 = f_{23}(x) + \frac{R \cdot b}{G_\alpha}\tau_y - \dot{\phi}_3 \tag{41}$$

which leads to the subsequent sliding-mode control law for ω

$$\tau_y = \frac{G_\alpha}{R \cdot b}[-f_{23}(x) + \dot{\phi}_3 - K_{s\omega 1}\operatorname{sgn}(S_\omega) - K_{s\omega 2}S_\omega] \tag{42}$$

where $K_{s\omega 1}$ and $K_{s\omega 2}$ are two real and positive constants. With the control law (42), (41) becomes

$$\dot{S}_\omega = -K_{s\omega 1}\operatorname{sgn}(S_\omega) - K_{s\omega 2}S_\omega \tag{43}$$

From (43), the fact that S_ω approaches zero in finite time is easily shown by choosing the Lyapunov function $V_1=0.5S_\omega^2$. It is worthwhile to note that S_ω must tend to zero as quickly as possible such that ω approaches the virtual control ϕ_3 in finite time.

2) Sliding-Mode Posture and Speed Torque Control Module

Next, move to derive the second sliding-mode posture and speed control law for the torque τ_p. With the aforementioned virtual controls of ω_α and v in (14) and (39), both backstepping errors are respectively expressed by

$$\eta_\alpha = \omega_\alpha - \phi_1 = \omega_\alpha - (-K_\alpha \alpha) \ , \quad \eta_v = v - \phi_2 \tag{44}$$

and their sliding surfaces are respectively defined by $S_\alpha = \eta_\alpha$ and $S_v = \eta_v$. Taking the time derivatives of both sliding surfaces and using the dynamic part in (11) yield

$$\begin{aligned} \dot{S}_\alpha &= \dot{\eta}_\alpha = \dot{\omega}_\alpha + K_\alpha \dot{\alpha} = f_{21}(\mathbf{x}) + g_{21}(\mathbf{x})\tau_p + K_\alpha \dot{\alpha} \\ \dot{S}_v &= \dot{\eta}_v = \dot{v} - \dot{\phi}_2 = f_{22}(\mathbf{x}) + g_{22}(\mathbf{x})\tau_p - \dot{\phi}_2 \end{aligned} \tag{45}$$

In (45), there is only one torque that has to stabilize both dynamics of S_α and S_v; this is so-called underactuated control problem. Among several approaches which have been useful for this kind of underactuated control problem, a hierarchical decoupling sliding-mode control method in [19] is used to accomplish the control goal, and proposed as follows;

$$S = r_1 S_v + r_2 S_\alpha \tag{46}$$

where r_1 and r_2 are two constants. Differentiating S obtains

$$\dot{S} = r_1\dot{S}_v + r_2\dot{S}_\alpha = r\left[{}_1 f_{22}(\mathbf{x}) + g_{22}(\mathbf{x})\tau_p - \dot{\phi}_2\right] + r_2\dot{S}_\alpha \tag{47}$$

Let the torque τ_p be

$$\tau_p = \frac{-r_1 f_{22}(\mathbf{x}) + r_1\dot{\phi}_2 - r_2\dot{S}_\alpha - K_{s1}\operatorname{sgn}(S) - K_{s2}S}{r_1 g_{22}(\mathbf{x})} \tag{48}$$

where K_{s1} and K_{s2} are two real and positive constants. Substituting (48) into (47) yields

$$\dot{S} = -K_{s1}\operatorname{sgn}(S) - K_{s2}S \tag{49}$$

Similar to the previous case, it is easy to show that S approaches zero in finite time by selecting the Lyapunov function V_2=0.5S^2. Worthy of mention is that $g_{22}(\mathbf{x})$ must be nonzero for every $\mathbf{x}$. Once the sliding surface S approaches zero, it is easy to show that $S_\alpha, S_v \to 0$ in finite time from the main result in [20].

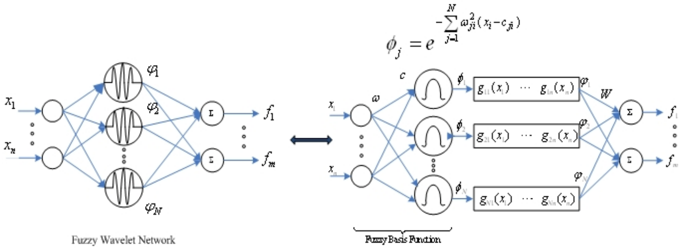


Fig. 2. Fuzzy wavelet network.

Theorem 2: Let r_1 and r_2 be two constants and $g_{22}(\mathbf{x})$ be always nonzero for every $\mathbf{x}$. Then both sliding-mode control laws (42) and (48) make the three backstepping errors, η_ω, η_v, and η_α, to approach zero in finite time.

IV. Intelligent Motion Control Using Fuzzy Wavelet Networks

This section will develop an intelligent motion controller using fuzzy wavelet networks (FWN). The controller synthesis procedure is composed of two steps. First, the fuzzy-wavelet-networks approximator is introduced to proximate the nonlinear terms in (42) and (48). Second, on-line learning rules of the motion controllers using FWNs are respectively established to achieve simultaneous tracking and stabilization.

A. Fuzzy-Wavelet-Network Approximator

Fig. 2 shows the schematic diagram of the fuzzy wavelet networks whose parameters are given as follows.

$$\varphi_j(x_i) = [1 - \omega_{ji}{}^2(x_i - c_{ji})^2]e^{\left\{-\omega_{ji}{}^2(x_i - c_{ji})^2\right\}} \tag{50}$$

where j=1, …, n, i=1, …, m, and the k^{th} output is described by

$$f_k = \sum_{j=1}^{N} w_{kj}\varphi_j(x) \quad k = 1, \cdots, m \tag{51}$$

Hence, (51) can be expressed in a vector-matrix form

$$\mathbf{f}(\mathbf{x}, \mathbf{c}, \boldsymbol{\omega}, \mathbf{W}) = \mathbf{W}^T\varphi(\mathbf{x}, \mathbf{c}, \boldsymbol{\omega}) \tag{52}$$

Next, use the fuzzy wavelet networks to approximate the nonlinear term $\mathbf{f}$. Let the ideal approximation result is

$$\mathbf{f} = \mathbf{W}^{*T}\varphi^*(\mathbf{x}, \mathbf{c}^*, \boldsymbol{\omega}^*) + \boldsymbol{\varepsilon}_f^* \tag{53}$$

where $\mathbf{W}^*$ is optimal weight vector; φ^* is optimal fuzzy wavelet basis function vector; $\boldsymbol{\varepsilon}_f^*$ is small and bounded error vector.

Assumption 1: The norm of optimal weight, $\|\mathbf{W}\|$, $\|\omega\|$,and $\|c\|$, are bounded, i.e.,

$$\left\|\mathbf{W}^*\right\| \le b_W < \infty, \left\|\boldsymbol{\omega}^*\right\| \le b_\omega < \infty \text{ and } \left\|\mathbf{c}^*\right\| \le b_c < \infty \qquad (54)$$

The actual fuzzy wavelet network estimate, $\hat{\mathbf{f}}$, can be represented by

$$\hat{\mathbf{f}} = \hat{\mathbf{W}}^T \hat{\varphi}(\mathbf{x}, \hat{\mathbf{c}}, \hat{\boldsymbol{\omega}}) \qquad (55)$$

where $\hat{\mathbf{W}}, \hat{\varphi}, \hat{c}, \hat{\omega}$ are estimates of $\mathbf{W}^*, \varphi^*, c^*, \omega^*$ in the fuzzy wavelet network. By defining $\tilde{\mathbf{W}} = \mathbf{W}^* - \hat{\mathbf{W}}$, $\tilde{\varphi} = \varphi^* - \hat{\varphi}$, $\tilde{\omega} = \omega^* - \hat{\omega}$, $\tilde{c} = c^* - \hat{c}$, (53) can be rewritten as

$$\begin{aligned} \mathbf{f} &= \mathbf{W}^{*T} \varphi^*(x, c^*, \omega^*) + \boldsymbol{\varepsilon}_f^* = (\tilde{\mathbf{W}} + \hat{\mathbf{W}})^T (\tilde{\varphi} + \hat{\varphi}) + \boldsymbol{\varepsilon}_f^* \\ &= \hat{\mathbf{f}} + \hat{\mathbf{W}}^T \tilde{\varphi} + \tilde{\mathbf{W}}^T \hat{\varphi} + \tilde{\mathbf{W}}^T \tilde{\varphi} + \boldsymbol{\varepsilon}_f^* \end{aligned} \qquad (56)$$

Let

$$\boldsymbol{\varepsilon} = \tilde{\mathbf{W}}^T \tilde{\varphi} + \boldsymbol{\varepsilon}_f^* \qquad (57)$$

Then

$$\mathbf{f} = \hat{\mathbf{f}} + \tilde{\mathbf{W}}^T \hat{\varphi} + \hat{\mathbf{W}}^T \tilde{\varphi} + \boldsymbol{\varepsilon} \qquad (58)$$

Furthermore, $\tilde{\varphi} = \varphi^* - \hat{\varphi}$ can be expanded by Taylor series as

$$\tilde{\varphi} = \frac{\partial \tilde{\varphi}}{\partial \hat{\omega}} \tilde{\omega} + \frac{\partial \tilde{\varphi}}{\partial \hat{c}} \tilde{c} + H = \mathbf{A}\tilde{\omega} + \mathbf{B}\tilde{c} + \mathbf{H} \qquad (59)$$

where

$$\mathbf{A} = \frac{\partial \tilde{\varphi}}{\partial \hat{\omega}} = \frac{\partial(\varphi^* - \hat{\varphi})}{\partial \hat{\omega}} = -\frac{\partial \hat{\varphi}}{\partial \hat{\omega}} = -\left[\frac{\partial \hat{\varphi}_1}{\partial \hat{\omega}} \;\; \frac{\partial \hat{\varphi}_2}{\partial \hat{\omega}} \;\; \ldots \;\; \frac{\partial \hat{\varphi}_N}{\partial \hat{\omega}}\right]_{N \times N}^T \qquad (60)$$

For *j=1,…n, i= 1,…,m, where*

$$\hat{\omega} = \left[\hat{\omega}_{11} \;\; \hat{\omega}_{12} \;\; \ldots \hat{\omega}_{1m} \;\; \cdots \hat{\omega}_{n1} \;\; \hat{\omega}_{n2} \;\; \ldots \hat{\omega}_{nm}\right]^T \qquad (61)$$

$$\frac{\partial \hat{\varphi}_j}{\partial \hat{\omega}} = \left[\underbrace{0....0}_{(j-1)\times n} \;\; \left(\partial \varphi_j / \partial \omega_{j1}\right) \;\; \cdots \;\; \left(\partial \varphi_j / \partial \omega_{jn}\right) \;\; \underbrace{0....0}_{(N-1)\times n}\right] \qquad (62)$$

$$\frac{\partial \varphi_i}{\partial \omega_{ji}} = -\frac{[2 - \omega_{ji}^2 (x_i - c_{ji})^2][-2\omega_{ji}(x_i - c_{ji})^2]}{1 - \omega_{ji}^2 (x_i - c_{ji})^2} \varphi_i(x_i) \qquad (63)$$

Similarly,

$$\mathbf{B} = \left.\frac{\partial \tilde{\varphi}}{\partial \hat{c}}\right|_{\substack{c=\hat{c} \\ \omega=\hat{\omega}}} = \frac{\partial(\varphi^* - \hat{\varphi})}{\partial \hat{c}} = -\frac{\partial \hat{\varphi}}{\partial \hat{c}} = -\left[\frac{\partial \hat{\varphi}_1}{\partial \hat{c}} \;\; \ldots \;\; \frac{\partial \hat{\varphi}_N}{\partial \hat{c}}\right]_{N \times N}^T \qquad (64)$$

$$\hat{c} = \left[\hat{c}_{11} \;\; \hat{c}_{12} \;\; \ldots \hat{c}_{1m} \;\; \cdots \hat{c}_{n1} \;\; \hat{c}_{n2} \;\; \ldots \hat{c}_{nm}\right]^T \qquad (65)$$

$$\frac{\partial \hat{\varphi}_j}{\partial \hat{c}} = \left[\underbrace{0....0}_{(j-1)\times n} \;\; \left(\partial \varphi_j / \partial c_{j1}\right) \;\; \cdots \;\; \left(\partial \varphi_j / \partial c_{jn}\right) \;\; \underbrace{0....0}_{(N-1)\times n}\right] \qquad (66)$$

$$\frac{\partial \varphi_i}{\partial c_{ji}} = -\frac{2\omega_{ji}(x_i - c_{ji})[2 - \omega_{ji}^2 (x_i - c_{ji})^2]}{1 - \omega_{ji}^2 (x_i - c_{ji})^2} \varphi_i(x_i) \qquad (67)$$

With (58), (51) becomes

$$\begin{aligned} \mathbf{f} &= \hat{\mathbf{f}} + \hat{\mathbf{W}}^T(\mathbf{A}\tilde{\omega} + \mathbf{B}\tilde{c} + \mathbf{H}) + \tilde{\mathbf{W}}^T \hat{\varphi} + \boldsymbol{\varepsilon} \\ &= \hat{\mathbf{f}} + \hat{\mathbf{W}}^T \mathbf{A}\tilde{\omega} + \hat{\mathbf{W}}^T \mathbf{B}\tilde{c} + \tilde{\mathbf{W}}^T \hat{\varphi} + \hat{\mathbf{W}}^T \mathbf{H} + \boldsymbol{\varepsilon} \end{aligned} \qquad (68)$$

Let $\mathbf{h} = \hat{\mathbf{W}}^T \mathbf{H} + \boldsymbol{\varepsilon}$, (68) can be rewritten by

$$\mathbf{f} = \hat{\mathbf{f}} + \hat{\mathbf{W}}^T \mathbf{A}\tilde{\omega} + \hat{\mathbf{W}}^T \mathbf{B}\tilde{c} + \tilde{\mathbf{W}}^T \hat{\varphi} + \mathbf{h} \qquad (69)$$

which $\mathbf{h}$ has a bounded infinity-norm, i.e., $\left\|\mathbf{h}\right\|_\infty < \infty$.

B. Intelligent Adaptive Yaw Rate Controller Design Using FWN

This subsection intends to develop the intelligent yaw rate controller using FWN. In doing so, rewrite (57) as

$$\begin{aligned} C_y &= \frac{G_\alpha}{R \cdot b}[-f_{23}(\mathbf{x}) + \dot{\phi}_3 - K_{\eta w1}\operatorname{sgn}(S_\eta) - K_{\eta w2} S_\eta] \\ &= -f_y(\mathbf{x}) + \frac{G_\alpha}{R \cdot b}\dot{\phi}_3 - \frac{G_\alpha K_{\eta w1}}{R \cdot b}\operatorname{sgn}(S_\eta) - \frac{G_\alpha K_{\eta w2}}{R \cdot b} S_\eta \end{aligned} \qquad (70)$$

where $f_y(x) = \left[G_\alpha / (R \cdot b)\right]\left(f_{23}(x) - \dot{\phi}_3\right),\ 0 < \left|G_\alpha / (R \cdot b)\right| \le K_5$

To meet the reaching condition, one proposes the subsequent intelligent yaw rate controller by

$$C_y = -\hat{f}_y(\mathbf{x}) + [-K_6 - g_y]\operatorname{sgn}(S_\eta), g_y \ge g_{y\max}, K_6 > 0 \qquad (71)$$

where $\hat{f}_y$ is the estimate of f_y using the FWN proposed in Section 4.1, namely that

$$f_y = \mathbf{W}_y^{*T} \boldsymbol{\Phi}_y^* + \varepsilon_y^* = \left[W_{y_1}^* \cdots W_{y_n}^*\right]\left[\Phi_{y_1}^* \ldots \Phi_{y_n}^*\right]^{\mathrm{T}} + \varepsilon_y^* \qquad (72)$$

and

$$\hat{f}_y = \hat{\mathbf{W}}_y{}^T \hat{\boldsymbol{\Phi}}_y = [\hat{W}_{y1} \cdots \hat{W}_{yn}][\hat{\Phi}_{y1} \;\; \cdots \;\; \hat{\Phi}_{yn}]^{\mathrm{T}} \qquad (73)$$

Moreover, by defining $\tilde{\mathbf{W}}_y = \mathbf{W}_y{}^* - \hat{\mathbf{W}}_y$, $\tilde{\boldsymbol{\Phi}}_y = \boldsymbol{\Phi}_y^* - \hat{\boldsymbol{\Phi}}_y$, one obtains

$$f_y = \hat{\mathbf{W}}_y^{\mathbf{T}} \hat{\boldsymbol{\Phi}}_y + \hat{\mathbf{W}}_y^{\mathbf{T}} \tilde{\boldsymbol{\Phi}}_y + \tilde{\mathbf{W}}_y^{\mathbf{T}} \widehat{\boldsymbol{\Phi}}_y + \tilde{\mathbf{W}}_y^{\mathbf{T}} \tilde{\boldsymbol{\Phi}}_y + \varepsilon_y^* \qquad (74)$$

In order to achieve on-line tuning of the FWN parameters, including the center vector $\mathbf{C}_y$=[c_{y11} c_{y21} c_{y31} … c_{y1n} c_{y2n} c_{y3n}]^T and the vector $\boldsymbol{\omega}_y$=[ω_{y1} ω_{y2} ω_{y3} … ω_{yn}]^T, the expansion of $\tilde{\Phi}$ is taken in a Taylor series as follows.

$$\boldsymbol{\Phi}_y = \frac{\partial \tilde{\boldsymbol{\Phi}}_y}{\partial \hat{\mathbf{C}}_y} \tilde{\mathbf{C}}_y + \frac{\partial \tilde{\boldsymbol{\Phi}}_y}{\partial \hat{\boldsymbol{\omega}}_y} \tilde{\boldsymbol{\omega}}_y + \mathbf{h}_y = \mathbf{A}_y \tilde{\mathbf{C}}_y + \mathbf{B}_y \tilde{\boldsymbol{\omega}}_y + \mathbf{h}_y \qquad (75)$$

where $\tilde{\mathbf{C}}_y = \mathbf{C}_y^* - \hat{\mathbf{C}}_y$; $\tilde{\boldsymbol{\omega}}_y = \boldsymbol{\omega}_y^* - \hat{\boldsymbol{\omega}}_y$; h_p is the vector containing higher order terms and satisfies $\left\|h_p\right\| \le b$. Substituting (74) into (75) gives

$$f_y = \hat{\mathbf{W}}_y{}^T \hat{\boldsymbol{\Phi}}_y + \hat{\mathbf{W}}_y{}^T (\mathbf{A}_y \tilde{\mathbf{C}}_y + \mathbf{B}_y \tilde{\boldsymbol{\omega}}_y) + \tilde{\mathbf{W}}_y{}^T \hat{\boldsymbol{\Phi}}_y + \varepsilon_y \qquad (76)$$

where $\varepsilon_y = \tilde{\mathbf{W}}_y{}^T \mathbf{h}_y + \tilde{\mathbf{W}}_y{}^T \tilde{\boldsymbol{\Phi}}_y + \varepsilon_y^*$ and ε is assumed to satisfies $\left|\varepsilon_y\right| < g_{y\max}$.

Substituting C_y into $\dot{S}_\eta$ gives

$$
\begin{aligned}
\dot{S}_\eta &= \frac{R\cdot b}{G_\alpha}[C_y + f_y] \\
&= \frac{R\cdot b}{G_\alpha}[f_y - \hat{f}_y + [-K_6 - g_y]\operatorname{sgn}(S_\eta)] \qquad (77) \\
&= \frac{R\cdot b}{G_\alpha}[\hat{\mathbf{W}}_{\mathbf{y}}{}^T\tilde{\mathbf{\Phi}}_{\mathbf{y}} + \hat{\mathbf{W}}_{\mathbf{y}}{}^T\hat{\mathbf{\Phi}}_{\mathbf{y}} + \varepsilon_y + (-K_6 - g_y)\operatorname{sgn}(S_\eta)]
\end{aligned}
$$

and

$$
\begin{aligned}
S_\eta\dot{S}_\eta &\le \frac{R\cdot b}{G_\alpha}\begin{bmatrix} \hat{\mathbf{W}}_{\mathbf{y}}{}^T((\mathbf{A}_{\mathbf{y}}\tilde{\mathbf{C}}_{\mathbf{y}} + \mathbf{B}_{\mathbf{y}}\tilde{\boldsymbol{\omega}}_{\mathbf{y}}) + \tilde{\mathbf{W}}_{\mathbf{y}}{}^{\mathbf{T}}\hat{\mathbf{\Phi}}_{\mathbf{y}})S_\eta \\ +|\varepsilon_y||S_\eta| + (-K_6 - g_y)|S_\eta| \end{bmatrix} \\
&\le \frac{R\cdot b}{G_\alpha}\begin{bmatrix} \hat{\mathbf{W}}_{\mathbf{y}}{}^{\mathbf{T}}((\mathbf{A}_{\mathbf{y}}\tilde{\mathbf{C}}_{\mathbf{y}} + \mathbf{B}_{\mathbf{y}}\tilde{\boldsymbol{\omega}}_{\mathbf{y}}) + \tilde{\mathbf{W}}_{\mathbf{y}}{}^{\mathbf{T}}\hat{\mathbf{\Phi}}_{\mathbf{y}})S_\eta + \\ -(g_y - g_{y\max})|S_\eta| - K_6|S_\eta| \end{bmatrix} \qquad (78) \\
&\le \Delta_y\left\{\left[\hat{\mathbf{W}}_{\mathbf{y}}{}^{\mathbf{T}}(\mathbf{A}_{\mathbf{y}}\tilde{\mathbf{C}}_{\mathbf{y}} + \mathbf{B}_{\mathbf{y}}\tilde{\boldsymbol{\omega}}_{\mathbf{y}}) + \tilde{\mathbf{W}}_{\mathbf{y}}{}^{\mathbf{T}}\hat{\mathbf{\Phi}}_{\mathbf{y}}\right]S_\eta - K_6|S_\eta|\right\}
\end{aligned}
$$

To obtain the parameter updating rules for $\hat{W}_y$, $\hat{C}_y$, and $\hat{\omega}_y$, we have

$$
V_3 = \frac{1}{2}S_\eta{}^2 + \frac{\tilde{\mathbf{W}}_{\mathbf{y}}{}^T\tilde{\mathbf{W}}_{\mathbf{y}}}{2\lambda_{W_y}} + \frac{\tilde{\mathbf{C}}_{\mathbf{y}}{}^T\tilde{\mathbf{C}}_{\mathbf{y}}}{2\lambda_{C_y}} + \frac{\tilde{\boldsymbol{\omega}}_{\mathbf{y}}{}^T\tilde{\boldsymbol{\omega}}_{\mathbf{y}}}{2\lambda_{\omega_y}} \qquad (79)
$$

Taking the time derivative of V_3 gives

$$
\begin{aligned}
\dot{V}_3 &= S_\eta\dot{S}_\eta + \frac{\tilde{\mathbf{W}}_{\mathbf{y}}{}^T}{\lambda_{W_y}}\left(-\dot{\hat{\mathbf{W}}}_y\right) + \frac{\tilde{\mathbf{C}}_{\mathbf{y}}{}^T}{\lambda_{C_y}}\left(-\dot{\hat{\mathbf{C}}}_y\right) + \frac{\tilde{\boldsymbol{\omega}}_{\mathbf{y}}{}^T}{\lambda_{\omega_y}}\left(-\dot{\hat{\boldsymbol{\omega}}}_{\mathbf{y}}\right) \\
&= -K_6\Delta_2|S_\eta| + \tilde{\mathbf{W}}_{\mathbf{y}}{}^T\left(-\frac{\dot{\hat{\mathbf{W}}}_{\mathbf{y}}}{\lambda_{W_y}} + \Delta_2\hat{\Phi}_y S_\eta\right) \qquad (80) \\
&\quad + \tilde{\mathbf{C}}_{\mathbf{y}}{}^T\left(-\frac{\dot{\hat{\mathbf{C}}}_{\mathbf{y}}}{\lambda_{C_y}} + \mathbf{A}_{\mathbf{y}}{}^T\hat{W}_y\Delta_2\right) + \tilde{\boldsymbol{\omega}}_{\mathbf{y}}{}^T\left(-\frac{\dot{\hat{\boldsymbol{\omega}}}_{\mathbf{y}}}{\lambda_{\omega_y}} + \mathbf{B}_{\mathbf{y}}\hat{\mathbf{W}}_{\mathbf{y}}\Delta_2\right)
\end{aligned}
$$

Then the parameter updating rules can be selected as

$$
\begin{cases}
\dot{\hat{\mathbf{W}}}_y = \lambda_{W_y}\Delta_2\hat{\Phi}_y S_\eta \\
\dot{\hat{\mathbf{C}}}_y = \lambda_{C_y}\mathbf{A}_{\mathbf{y}}{}^T\hat{\mathbf{W}}_{\mathbf{y}}\Delta_2 \\
\dot{\hat{\boldsymbol{\omega}}}_{\mathbf{y}} = \lambda_{\omega_y}\mathbf{B}_{\mathbf{y}}{}^T\hat{\mathbf{W}}_{\mathbf{y}}\Delta_2
\end{cases} \qquad (81)
$$

where $\Delta_y = \frac{R\cdot b}{G_\alpha}$. Then $\dot{V}_3 \le -K_6\Delta_y|S_\eta| \le 0$ and it implies that $S_\eta \to 0$ in finite time.

Theorem 3: Consider the subsystem (15) with the proposed intelligent adaptive yaw rate control law (70) with the parameter updating laws (81). Then $\omega \to \phi_3$ in finite time.

C. *Intelligent Adaptive Posture and Speed Controllers Design Using FWN*

This subsection is aimed at developing an intelligent adaptive posture and speed control law for C_θ, it is necessary to rewrite (48) in the following form.

$$
\begin{aligned}
C_\theta &= \frac{1}{r_1 g_{22}}\left[-r_1 f_{22}(\mathbf{x}) + r_1\dot{\phi}_2 - r_2\dot{S}_a - K_{S1}sgn(S_1) - K_{S2}S_1\right] \\
&= -f_p(\mathbf{x}) - \frac{K_{S1}}{r_1 g_{22}(\mathbf{x})}sgn(S_1) - \frac{K_{S2}}{r_1 g_{22}(\mathbf{x})}S_1
\end{aligned} \qquad (82)
$$

where $f_p(\mathrm{x}) = \frac{f_{22}(\mathrm{x})}{g_{22}(\mathrm{x})} - \frac{1}{g_{22}(\mathbf{x})}\dot{\phi}_2 + \frac{r_2}{r_1 g_{22}(\mathbf{x})}\dot{S}_a$. Moreover, $\left|\frac{1}{g_{22}(\mathbf{x})}\right| \le K_1$, $\left|\frac{r_2}{r_1 g_{22}(\mathbf{x})}\right| \le K_2$, $\left|\frac{K_{S1}}{r_1 g_{22}(\mathbf{x})}\right| \le K_{\theta 1}$, $\left|\frac{K_{S2}}{r_1 g_{22}(\mathbf{x})}\right| \le K_{\theta 2}$. Since the function f_p in (82) can be on-line learned by the FWN, it is good to propose the following adaptive control

$$
C_\theta = -\hat{f}_p(\mathbf{x}) - \frac{K_{S1}}{r_1 g_{22}(\mathbf{x})}sgn(S_1) - \frac{K_{S2}}{r_1 g_{22}(\mathbf{x})}S_1 \qquad (83)
$$

where K_{S1}, K_{S2}>0, $\hat{f}_p$ is the estimate of f_p using the FWN proposed in Section 4-*A*, namely that

$$
f_p = \mathbf{W}_p^{*T}\mathbf{\Phi}_p^* + \varepsilon_p^* = [W_{p1}^* \cdots W_{pn}^*][\Phi^*_{p1} ... \Phi^*_{pn}]^T + \varepsilon_p^* \qquad (84)
$$

and

$$
\hat{f}_p = \hat{\mathbf{W}}_{\mathbf{p}}{}^T\hat{\mathbf{\Phi}}_{\mathbf{p}} = [\hat{W}_{p1} \cdots \hat{W}_{pn}][\hat{\Phi}_{p1} \quad \cdots \quad \hat{\Phi}_{pn}]^{\mathrm{T}} \qquad (85)
$$

Moreover, by defining $\tilde{\mathbf{W}}_{\mathbf{p}} = \mathbf{W}_{\mathbf{p}}{}^* - \hat{\mathbf{W}}_{\mathbf{p}}$, $\tilde{\mathbf{\Phi}}_{\mathbf{p}} = \mathbf{\Phi}^*_{\mathbf{p}} - \hat{\mathbf{\Phi}}_{\mathbf{p}}$, one obtains

$$
f_p = \hat{\mathbf{W}}_{\mathbf{p}}^T\hat{\mathbf{\Phi}}_{\mathbf{p}} + \hat{\mathbf{W}}_{\mathbf{p}}^T\tilde{\mathbf{\Phi}}_{\mathbf{p}} + \tilde{\mathbf{W}}_{\mathbf{p}}^T\hat{\mathbf{\Phi}}_{\mathbf{p}} + \tilde{\mathbf{W}}_{\mathbf{p}}^T\tilde{\mathbf{\Phi}}_{\mathbf{p}} + \varepsilon_p^* \qquad (86)
$$

In order to achieve on-line tuning of the FWN parameters, including the center vector $\mathbf{C}_p$=[c_{p11} c_{p21} c_{p31} … c_{p1n} c_{p2n} c_{p3n}]^T and the vector $\boldsymbol{\omega}_p$=[ω_{p1} ω_{p2} ω_{p3} … ω_{pn}]^T, the expansion of $\tilde{\Phi}_p$ is taken in a Taylor series as follows.

$$
\tilde{\mathbf{\Phi}}_{\mathbf{p}} = \frac{\partial\tilde{\mathbf{\Phi}}_{\mathbf{p}}}{\partial\hat{\mathbf{C}}_{\mathbf{p}}}\tilde{\mathbf{C}}_{\mathbf{p}} + \frac{\partial\tilde{\mathbf{\Phi}}_{\mathbf{p}}}{\partial\hat{\boldsymbol{\omega}}_{\mathbf{p}}}\tilde{\boldsymbol{\omega}}_{\mathbf{p}} + \mathbf{h}_{\mathbf{p}} = \mathbf{A}_{\mathbf{p}}\tilde{\mathbf{C}}_{\mathbf{p}} + \mathbf{B}_{\mathbf{p}}\tilde{\boldsymbol{\omega}}_{\mathbf{p}} + \mathbf{h}_{\mathbf{p}} \qquad (87)
$$

where $\tilde{\mathbf{C}}_{\mathbf{p}} = \mathbf{C}^*_{\mathbf{p}} - \hat{\mathbf{C}}_{\mathbf{p}}$; $\tilde{\boldsymbol{\omega}}_{\mathbf{p}} = \boldsymbol{\omega}^*_{\mathbf{p}} - \hat{\boldsymbol{\omega}}_{\mathbf{p}}$; h is the vector containing higher order terms and satisfies $\|\mathbf{h}_{\mathbf{p}}\| \le b$. Substituting (87) into (86) gives

$$
f_p = \hat{\mathbf{W}}_{\mathbf{p}}{}^T\hat{\mathbf{\Phi}}_{\mathbf{p}} + \hat{\mathbf{W}}_{\mathbf{p}}{}^T(\mathbf{A}_{\mathbf{p}}\tilde{\mathbf{C}}_{\mathbf{p}} + \mathbf{B}_{\mathbf{p}}\tilde{\boldsymbol{\omega}}_{\mathbf{p}}) + \tilde{\mathbf{W}}^T\hat{\mathbf{\Phi}}_{\mathbf{p}} + \varepsilon_p \qquad (88)
$$

where $\varepsilon_p = \hat{\mathbf{W}}_{\mathbf{p}}{}^T\mathbf{h}_{\mathbf{p}} + \tilde{\mathbf{W}}_{\mathbf{p}}{}^T\tilde{\mathbf{\Phi}}_{\mathbf{p}} + \varepsilon_p^*$ and ε is assumed to satisfies $|\varepsilon_p| < g_{p\max}$.

Substituted the proposed controller into $\dot{S}_1$

$$\begin{aligned}\dot{S}_1 &= r_1 f_{22}(\mathbf{x}) + r_1 g_{22}(\mathbf{x}) C_\theta - r_1\dot{\phi}_2 + r_2 \dot{S}_\alpha \\ &= r_1 g_{22}(\mathbf{x})\left[C_\theta + f_p(\mathbf{x})\right]\end{aligned} \tag{89}$$

Moreover, it is easy to rewrite the following two terms:

$$\begin{aligned}&C_\theta + \frac{f_{22}(\mathbf{x})}{g_{22}(\mathbf{x})} - \frac{1}{g_{22}(\mathbf{x})}\dot{\phi}_2 + \frac{r_2}{r_1 g_{22}(\mathbf{x})}\dot{S}_\alpha \\ &= -\hat{f}_p(\mathbf{x}) - \frac{K_{S1}}{r_1 g_{22}(\mathbf{x})} sgn(S_1) - \frac{K_{S2}}{r_1 g_{22}(\mathbf{x})} S_1 + f_p\end{aligned} \tag{90}$$

With (88), one obtains

$$f_p(\mathbf{x}) - \hat{f}_p(\mathbf{x}) = \hat{\mathbf{W}}_p^T\left(\mathbf{A_p}\tilde{\mathbf{C}}_\mathbf{p} + \mathbf{B_p}\tilde{\boldsymbol{\omega}}_\mathbf{p}\right) + \tilde{\mathbf{W}}_p^T\hat{\boldsymbol{\Phi}}_\mathbf{p} + \varepsilon_p \tag{91}$$

and

$$\begin{aligned}\left[C_\theta + f_p(\mathbf{x})\right]S_1 &= \begin{bmatrix}\tilde{\mathbf{W}}_\mathbf{p}{}^T\left(\mathbf{A_p}\tilde{\mathbf{C}}_\mathbf{p} + \mathbf{B_p}\tilde{\mathbf{W}}_\mathbf{p}\right) \\ +\tilde{\mathbf{W}}_p^T\hat{\boldsymbol{\Phi}}_\mathbf{p} + \mathbf{W}_p^T\tilde{\boldsymbol{\Phi}}_\mathbf{p} + \varepsilon_p\end{bmatrix}S_1 \\ &\quad - \frac{K_{S1}}{r_1 g_{22}(\mathbf{x})}\left|S_1\right| - \frac{K_{S2}}{r_1 g_{22}(\mathbf{x})}S_1{}^2\end{aligned} \tag{92}$$

where $\left|\varepsilon_p\right| \le g_{p\max}, \varepsilon_p = \tilde{\mathbf{W}}_p^T\tilde{\boldsymbol{\Phi}}_\mathbf{p} + \hat{\mathbf{W}}_p^T\mathbf{h_p} + \varepsilon_p{}^*$

Using the inequality $A + B \ge A - \left|B\right|$, one obtains

$$\begin{aligned}\left[C_\theta + f_p(\mathbf{x})\right]S_1 &\le S_1\left[\hat{\mathbf{W}}_\mathbf{p}{}^T(\mathbf{A_p}\tilde{\mathbf{C}}_\mathbf{p} + \mathbf{B_p}\tilde{\boldsymbol{\omega}}_\mathbf{p}) + \tilde{\mathbf{W}}_\mathbf{p}{}^T\widehat{\boldsymbol{\Phi}}_\mathbf{p}\right] \\ &\quad - \left|\varepsilon_p\right|\left|S_1\right| - \frac{K_{S1}}{r_1 g_{22}(\mathbf{x})}\left|S_1\right| - \frac{K_{S2}}{r_1 g_{22}(\mathbf{x})}S_1{}^2 \\ &\le S_1\left[\hat{\mathbf{W}}_\mathbf{p}{}^T(\mathbf{A_p}\tilde{\mathbf{C}}_\mathbf{p} + \mathbf{B_p}\tilde{\boldsymbol{\omega}}_\mathbf{p}) + \tilde{\mathbf{W}}_\mathbf{p}{}^T\widehat{\boldsymbol{\Phi}}_\mathbf{p}\right] \\ &\quad - \left(\frac{K_{S1}}{r_1 g_{22}(\mathbf{x})} + g_{p\max}\right)\left|S_1\right| - \frac{K_{S2}}{r_1 g_{22}(\mathbf{x})}S_1{}^2\end{aligned} \tag{93}$$

where $\tilde{\mathbf{C}}_\mathbf{p} = \mathbf{C_p}{}^* - \hat{\mathbf{C}}_\mathbf{p}$, $\tilde{\boldsymbol{\omega}}_\mathbf{p} = \boldsymbol{\omega}_\mathbf{p}{}^* - \hat{\boldsymbol{\omega}}_\mathbf{p}$, $\tilde{\mathbf{W}}_\mathbf{p} = \mathbf{W_p}{}^* - \hat{\mathbf{W}}_\mathbf{p}$, Since $r_1 > 0$ and $r_1 g_{22}(x) < 0$, one obtains

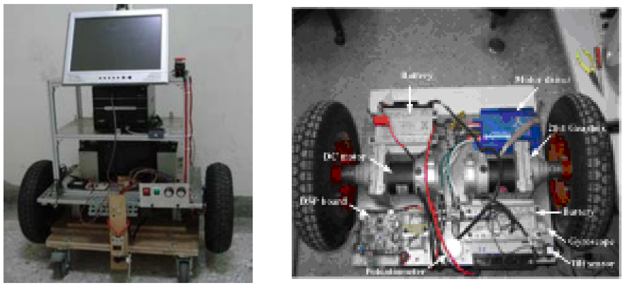
$$\begin{aligned}S_1\dot{S}_1 &\le r_1 g_{22}(x)\left\{\mathrm{S}_1\left[\hat{\mathbf{W}}_\mathbf{p}{}^T\left(\mathbf{A_p}\tilde{\mathbf{C}}_\mathbf{p} + \mathbf{B_p}\tilde{\boldsymbol{\omega}}_\mathbf{p}\right) + \tilde{\mathbf{W}}_\mathbf{p}{}^T\hat{\boldsymbol{\Phi}}_\mathbf{p}\right]\right\} \\ &\quad - \left(K_{S1} + r_1 g_{22}(\mathbf{x}) g_{p\max}\right)\left|S_1\right| - K_{S2}S_1{}^2\end{aligned} \tag{94}$$

Next, move to find the parameter updating laws for $\hat{g}_{p\max}$, $\hat{K}_1$, $\hat{K}_2$, $\hat{\mathbf{W}}_\mathbf{p}$, $\hat{\mathbf{C}}_\mathbf{p}$, $\hat{\boldsymbol{\omega}}_\mathbf{p}$. In doing so, we propose

$$V_4 = \frac{1}{2}S_1{}^2 + \frac{\tilde{\mathbf{C}}_\mathbf{p}{}^T\tilde{\mathbf{C}}_\mathbf{p}}{2\lambda_{C_p}} + \frac{\tilde{\boldsymbol{\omega}}_\mathbf{p}{}^T\tilde{\boldsymbol{\omega}}_\mathbf{p}}{2\lambda_{C_p}} + \frac{\tilde{\mathbf{W}}_\mathbf{p}{}^T(\tilde{\mathbf{W}}_\mathbf{p})}{2\lambda_{w_p}} \tag{95}$$

Which leads to

$$\begin{aligned}\dot{V}_4 &= \tilde{C}_p\mathbf{A}_p^T\hat{\mathbf{W}}_p S_1\Delta_p + \tilde{\boldsymbol{\omega}}_p^T\mathbf{B}_p^T\hat{\mathbf{W}}_p S_1\Delta_p + \tilde{\mathbf{W}}_p^T\left(\hat{\boldsymbol{\Phi}}_p\right)\Delta_p \\ &\quad - \left(K_{S1} + \Delta_p g_{p\max}\right)\left|S_1\right| - K_{S2}S_1{}^2 \\ &\quad + \frac{\tilde{\mathbf{C}}_p^T}{\lambda_{C_p}}\left(-\dot{\hat{\mathbf{C}}}_p\right) + \frac{\tilde{\boldsymbol{\omega}}_p^T}{\lambda_{\omega_p}}\left(-\dot{\hat{\boldsymbol{\omega}}}_p\right) + \frac{\tilde{\mathbf{W}}_p^T}{\lambda_{W_p}}\left(-\dot{\hat{\mathbf{W}}}_p\right)\end{aligned}$$



(a) (b)

Fig. 3. (a) Experimental wheeled inverted pendulum. (b) Bottom of the wheeled inverted pendulum.

$$\begin{aligned}\dot{V}_4 &= -\left(K_{S1} + \Delta_p g_{p\max}\right)\left|S_1\right| - K_{S2}S_1{}^2 \\ &\quad + \tilde{\mathbf{W}}_p^T\left[-\frac{\dot{\hat{\mathbf{W}}}_p}{\lambda_{W_p}} + \left(\hat{\boldsymbol{\Phi}}_p\right)\Delta_p\right] + \tilde{\mathbf{C}}_p^T\left[-\frac{\dot{\hat{\mathbf{C}}}_p}{\lambda_{C_p}} + \mathbf{A}_p^T\hat{\mathbf{W}}_p S_1\Delta_p\right] \\ &\quad + \tilde{\mathbf{W}}_p^T\left[-\frac{\dot{\hat{\mathbf{W}}}_p}{\lambda_{W_p}} + \mathbf{B}_p^T\hat{\mathbf{W}}_p S_1\Delta_p\right]\end{aligned} \tag{96}$$

where $\Delta_p = r_1 g_{22}(x)$. Then the parameter adjustment rules can be chosen by

$$\begin{cases}\dot{\hat{\mathbf{W}}}_\mathbf{p} = \lambda_{W_p}\Delta_p\hat{\boldsymbol{\Phi}}_\mathbf{p} \\ \dot{\hat{\mathbf{C}}}_\mathbf{p} = \lambda_{C_p}\mathbf{A_p}{}^T\hat{\mathbf{W}}_\mathbf{p}S_1\Delta_p \\ \dot{\hat{\boldsymbol{\omega}}}_\mathbf{p} = \lambda_{\omega_p}\mathbf{B_p}{}^T\hat{\mathbf{W}}_\mathbf{p}S_1\Delta_p\end{cases} \tag{97}$$

Then $\dot{V}_4 = -\left(K_{S1} + \Delta_p g_{p\max}\right)\left|S_1\right| - K_{S2}S_1{}^2 \le -\left(K_{S1} + \Delta_p g_{p\max}\right)\left|S_1\right| \le 0$, if $\left(K_{S1} + \Delta_p g_{p\max}\right) > 0$, implies $S_1 \to 0$ in finite time

Theorem 4: Consider the subsystem (15) with the proposed intelligent adaptive posture and speed control law (82) with the parameter updating laws (97). Then $\theta_p \to \phi_3, v \to \phi_2$ as $t \to \infty$.

V. Simulations and Discussion

In the section, simulations and experimental results are performed to examine the performance and merit of the proposed control method on an experimental wheeled inverted pendulum. After a brief description of the experimental wheeled inverted pendulum, four simulations are performed to examine the feasibility and effectiveness of the proposed motion control method. With those tuned control parameters obtained from simulation results, two experimental results are then conducted to show the performance and applicability of the proposed method.

A. Brief Description of the Experimental Wheeled Inverted Pendulum

Fig. 3 displays the picture of the experimental wheeled inverted pendulum. As shown in Fig. 3 (b), this wheeled inverted pendulum is composed of two 24V DC motors with gearbox and two stamped steel wheels with 16” tires, two 12-volt sealed rechargeable lead-acid batteries in series, and

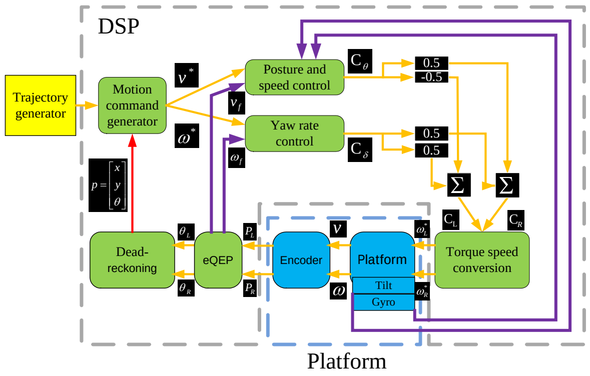


Fig. 4. Block diagram of the control architecture for the vehicle.

TABLE 2. ALL THE PENDULUM'S PARAMETERS USED FOR SIMULATIONS.

Parameter [unit]	Value	Parameter [unit]	Value
M_b [kg]	35	L [m]	0.45
M_w [kg]	5	b [m]	0.2
I_{xx} [$kg{\cdot}m^2$]	2.1073	R [m]	0.25
I_{yy} [$kg{\cdot}m^2$]	1.8229	C_z [m]	0.25
I_{zz} [$kg{\cdot}m^2$]	0.649	G [m/sec^2]	9.8
I_{wa} [$kg{\cdot}m^2$]	0.1563	$\bar{L}$ [m]	0.45
I_{wd} [$kg{\cdot}m^2$]	0.0781		

two motor drivers. The system adopts a digital signal processor (DSP) TI 320F28335 from Texas Instrument as a main controller, one gyroscope and one tilt sensor as the two sensors. The DSP mounted on the robot platform controls the posture and linear speed of the platform by using one gyroscope and one tilt sensor which are employed for measuring the rate and the angle of the inclination of the platform. In addition, the DSP-based controller also provides controls for the yaw rate of the platform using one high-performance optical gyroscope. The two optical encoders installed on the driving wheels are employed to achieve dead-reckoning calculation of the platform.

Fig. 4 depicts the block diagram of the control architecture for the robot. In Fig. 3, there are three main control modules: the virtual motion command generator, slide-mode posture and speed control module, and, and slide-mode yaw rate control module. The virtual motion command generator provides ideal linear and angular speed commands for the speed and yaw rate control modules for steering the robot to exactly follow the desired trajectories. The posture and speed control module is responsible for achieving speed following as well as maintaining the robot posture. The yaw rate control module is aimed at tracking the desired yaw rate commands from the virtual motion command generator.

B. Simulations and Discussion

This subsection presents simulation results of STS controllers using MATLAB/Simulink under three cases: stabilization, straight line tracking, and circular tracking and. Table 2 lists the coefficients of the wheeled inverted pendulum used for simulations. In the simulations, the robot got started at the initial posture (0 m, 0 m, π/6 rad); the reference trajectories for trajectory tracking and the reference configurations for stabilization are assumed to start at (2 m, 2 m, π/4 rad) such that the initial values of the auxiliary signals are w(0)=1.5348, $\tilde{z}_1(0)=0.5,\ \tilde{z}_2(0)=0.9912.$ Therefore, the controller's parameters for the virtual motion command generator are set by k=0.25, k_1=1 and k_2=0.25, α_1=0.2, and

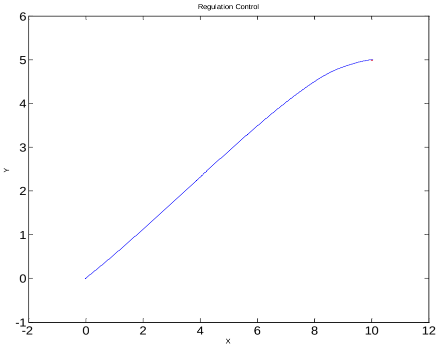


Fig. 5. Simulation result of the overall motion control system for regulation.

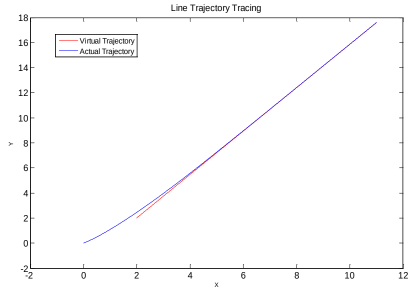


Fig. 6. Simulation result of the proposed controller for line trajectory tracing.

$$\alpha_0=\left(\left[(0-2)\cos(\pi/6)+(0-2)\sin(\pi/6)\right]^2+(\pi/6-\pi/4)^2\right)^{1/2}$$

$$z_d=\left[(0-2)\cos(\pi/6)+(0-2)\sin(\pi/6)\quad \pi/6-\pi/4\right]^T$$

thus satisfying the inequality min(kk_1, k_2)>α_1. Furthermore, the parameters for both sliding-mode control modules are given by r_1=1.0, r_2=0.2, K_α=1.0, K_{S1}=2.0, K_{S2}=2.0, $K_{S\omega1}$=2.0, and $K_{S\omega2}$=2.0, respectively. Worthy of mention is that the parameters of both sliding-mode control modules are easy to set up, but the three parameters of the virtual motion command generator are difficult to tune. All the three simulations were conducted using MATLAB/Simulink codes.

1) Stabilization

The first simulation was conducted to investigate the performance of the proposed STS motion control law (42) and (48) for stabilization. The initial pose of the wheeled inverted pendulum mobile robot was assumed at (x_0, y_0, θ_0)=(0m,0m,30°) and the final pose was set by (x, y, θ)=(10m, 5m, 0°). Fig. 5 depicts the simulation result of the pendulum moving from the initial posture to the desired posture. Through the simulation result, the mobile robot with the proposed STS controller is shown capable of reaching the desired postures with satisfactory performance.

2) Straight-Line Trajectory Tracking

The second simulation was to study straight-line trajectory tracking performance of the proposed STS motion control law. The initial pose of the robot was given by (x_0,y_0,θ_0)=(0m,0m,30°) and the desired line trajectory was described by $\left(x(t),\ y(t),\ \theta(t)\right)=\left(x_0+v_rt,\ y_0+v_rt,\ \theta_0+\omega_rt\right)$ where v_r = 0.6 m/s, ω_r = 0. Fig. 6 shows the simulation result of

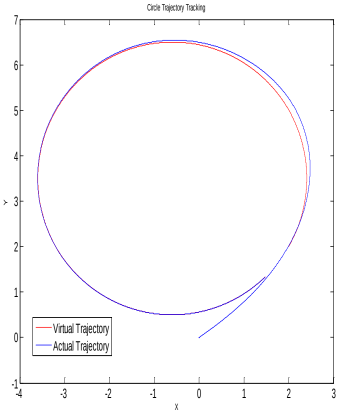


Fig. 7. Simulation result of the proposed controller for circle trajectory tracking.

the proposed control method. The result in Fig. 6 clearly indicates that the STS control laws are capable of steering the robot to follow the desired straight-line path.

3) *Circular Trajectory Tracking*

The third simulation for the STS motion controller was to steer the wheeled inverted pendulum mobile robot to follow a circular trajectory described by $\left(x(t),\ y(t),\ \theta(t)\right)=\left(x_0+r\cos\left(\omega t\right),\ y_0+r\sin\left(\omega t\right),\ \theta_0+\omega_r t\right)$ where $r=3\text{ m},\ \omega_r=0.2\text{ rad/s}$. Fig. 7 shows the simulation result of the wheeled inverted pendulum mobile robot with the proposed STS controller for circular trajectory tracking. The result indicates that the STS control laws perform well for steering the robot to exactly track the desired circular trajectory.

VI. Conclusions

This paper has presented a simultaneous tracking and stabilization (STS) method for wheeled inverted pendulums using backstepping and sliding-mode control techniques. The mathematical modeling of the pendulum has been decoupled into kinematic and dynamic parts in a state-space framework, and a unified STS controller has been synthesized in the two kinematic and dynamic levels. In the kinematic level, a STS controller has been constructed to achieve both goals of trajectory tracking and stabilization in a one unified control framework. In the dynamic level, the sliding mode control approaches have been employed to find the control torques acting on both wheels so as to accomplish out the overall control goals. Through simulations results, the proposed STS controllers have been shown useful and effective in providing appropriate control actions to steer the robot to achieve simultaneous tracking and stabilization. An interesting topic for future work might be to conduct experiments on a physical wheeled inverted pendulum.

References

[1] Y.Hosoda, S. Egawa, J. Tamamoto, K. Yamamoto, R.Nakamura and M.Togami "Basic design of human-symbiotic robot EMIEW," in *Pro. IEEE/RSJ International Conference on Intelligent Robots and Systems,* Beijing, China, pp.5079-5084, Oct. 9 - 15, 2006.

[2] S.C. Lin, *System design, modeling and control of self-balancing human transportation vehicles,* D.S. Thesis, Department of Electrical Engineering, National Chung Hsing University, Taichung, Taiwan, July 2008.

[3] M. Sasaki, N. Yanagihara, O. Matsumoto, and K. Komoriya, "Steering control of the personal riding-type wheeled mobile platform (PMP)," in *Proc. 2005 IEEE International Conference on Intelligent Robots and Systems*, pp.1697-1702, 2005.

[4] F. Grasser, A.D'Arrigo, and S. Colombi, "JOE: A Mobile, Inverted Pendulum," *IEEE Trans. Industrial Electronics*, vol.49, no.1, pp.107-114, February 2002.

[5] K. Pathak, J. Franch, and S. K. Agrawal, "Velocity and position control of a wheeled inverted pendulum by partial feedback linearization," *IEEE Transactions on* Robotics, vol.21, no.3, pp.505-513, June 2005.

[6] A. Salerno, and J. Angeles, "The control of semi-autonomous two-wheeled robots undergoing large payload-variations," in *Proc. ICRA'04*, vol.2, pp.1740-1745, Apr 26-May 1, 2004.

[7] Y.-S. Ha, S. Yuta, "Trajectory tracking control for navigation of the inverse pendulum type self-contained mobile robot," *Robotics and Autonomous Systems*, vol.17, pp. 65-80, 1996.

[8] E. A. Sisbot, L. F. Marin-Urias, R. Alami, and T. Simeon "A human aware mobile robot motion planner," *IEEE Transactions on* Robotics, vol.23, no.5, pp.874-883, October 2007.

[9] Y. H. Fan, *Motion control and planned navigation of a two-wheeled self-balancing mobile platform for human symbiotic robots*, M.S. Thesis, Department of Electrical Engineering, National Chung Hsing University, Taichung, Taiwan, July 2008.

[10] R. Siegwart and I R Nourbakhsh, *Introduction to autonomous mobile robots*, 1st Ed., Bradford, 2004.

[11] E. A. Sisbot, L. F. Marin-Urias, R. Alami, and T. Simeon "A mobile robot that performs human acceptable motions," in *Pro. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.1811-1816, October 2006.

[12] E. A. Sisbot, A. Clodic, L. F. Marin-U., M. Fontmarty, L. Brethes, and R. Alami, "Implementing a human-aware robot system," *IEEE International Symposium on Robot and Human Interactive Communication*, pp.727-732, September 2006.

[13] C. C. Tsai, C. K. Chan, and Y. H. Fan, "Planned Navigation of a self-balancing autonomous service robot," *International Conference on Advanced Robots and its Social Impacts*, Taipei, Taiwan, Aug. 2008.

[14] E. A. Sisbot, L. F. Marin-Urias, and R. Alami "Spatial reasoning for human robot interaction," in *Pro. IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, USA, pp.2281-2287, Nov. 2007.

[15] A. Shimada, and N. Hatakeyama, "High-speed motion control of wheeled inverted pendulum robots," *in Proc. 4th IEEE International Conference on Mechatronics*, pp.1-6, 8-10 May 2007.

[16] K. D. Do, Z. P. Jiang, and J. Pan, 2004a, "A Global Output-Feedback Controller for Simultaneous Tracking and Stabilization of Unicycle-Type Mobile Robots," *IEEE Transactions on Robotics and Automation*, Vol. 20, No. 3, pp. 589-594, 2004.

[17] K. D. Do, Z. P. Jiang, and J. Pan., "Simultaneous Tracking and Stabilization of Mobile Robots: an Adaptive Approach," *IEEE Transactions on Automatic Control*, Vol. 49, No.7, pp. 1147-1151, 2004.

[18] J. C. Lo and Y. H. Kuo, "Decoupled fuzzy sliding-mode control, "*IEEE Transactions on Fuzzy Systems*, vol. 6, no. 3, pp. 426-435. Aug. 1998.

[19] C. M. Lin and Y.J. Mon, "Decoupling Control by hierarchical fuzzy sliding-mode controller," *IEEE Transactions on Control System Technology*, vol. 13, no. 4, pp. 593-598, July 2005.

[20] W. Wang, X. D. Liu and J. Q. Yi, "Structure design of two types of sliding-mode controllers for a class of under-actuated mechanical systems," *IET Proceeding of Control Theory and Applications,* vol.1, no.1, pp. 163-172, Jan. 2007.

[21] D. W. C. Ho, P. A. Zhang and J. Xu, "Fuzzy wavelet networks for function learning," *IEEE Trans. on Fuzzy Systems*, vol. 9, no.1, pp.200-211, Feb. 2001.

[22] C. K. Lin, "Nonsingular terminal sliding mode control of robot manipulators using fuzzy wavelet networks," *IEEE Trans. on Fuzzy Systems,* vol. 14, no.6, pp.849-859, Dec. 2006.

[23] W. E. Dixon, D. M. Dawson, F. Zhang, and E. Zergeroglu, "Global exponential tracking control of a mobile robot system via a PE condition," *IEEE Transactions on Systems, Man and Cybernetics-part B*, vol. 30, no.1, pp. 129-142, Feb. 2000.

Ching-Chih Tsai received the Diplomat in Electrical Engineering from National Taipei Institute of Technology, Taipei, Taiwan, ROC, the MS degree in Control Engineering from National Chiao Tung University, Hsinchu, Taiwan, ROC and the Ph.D degree in Electrical Engineering from Northwestern University, Evanston, IL, USA, in 1981, 1986 and 1991, respectively. Currently, he is currently a Distinguished Professor in the Department of Electrical Engineering, National Chung-Hsing University, Taichung, Taiwan, where he served the Chairman in the Department of Electrical Engineering from 2012 to 2014. He is a Fellow of IEEE, IET and CACS.

Dr. Tsai served as the Chair, Taipei Chapter, IEEE Control Systems Society, from 2000 to 2003, and the Chair, Taipei Chapter, IEEE Robotics and Automation Society from 2005 to 2006. In 2007, he was the program chair of 2007 CACS international automatic conference sponsored by Taipei chapter, IEEE control systems society. In 2010, he served as the program co-chair of SICE 2010 annual conference in Taiwan, which was technically sponsored by IEEE CSS; in 2011, he served as the General Chair, 2011 International conference on service and interactive robotics; in 2012, he has served as the General Chair, 2012 International conference on Fuzzy Theory and Its Applications, the General Chair, 2012-2015 CACS International Automatic Control Conferences, and the General Chair, 2016-2017 International Conference on Advanced Robotics and Intelligent Systems. Dr. Tsai served the two-term President, Chinese Institute of Engineers in Central Taiwan, Taiwan from 2007 to 2011, and two-term President of Chinese Automatic Control Society from 2012 to 2015. Since 2008, he has been the Executive Directors in Boards of Government of thee professional associations, including Robotic Society of Taiwan, Taiwan Fuzzy Systems Association, and Taiwan Systems Association. He has served as the Chair, Taichung Chapter, IEEE Systems, Man, and Cybernetics Society since 2009, the Chair of IEEE SMC Technical Committee on intelligent learning in control systems since 2009, the President of Robotics Society of Taiwan since 2016, the steering committee of Asian Control Association since 2014, a BOG member of IEEE Nanotechnology council since 2012, the Vice President of International Fuzzy Systems Association since 2015, and a BOG member of the IEEE SMCS since 2017.

Dr. Tsai has published more than 500 technical papers, and seven patents in the fields of control theory, systems technology and applications. Web of Science has indexed his paper entitled "Adaptive Neural Network Control of a Self-Balancing Two-Wheeled Scooter" in the category Automation Control Systems, where the paper was ranked 408 out of 37607 articles (published between 2010 to 2014). Dr. Tsai is respectively the recipients of the Third Society Prize Paper Award from IEEE Industry Application Society in 1998, the Outstanding Automatic Control Engineering Award in 2008 from Chinese Automatic Control Society (CACS), and the Outstanding Engineering Professor Award in 2009 from the Chinese Institute of Engineers in 2009, the IEEE Most Active SMC Technical Committee (TC) Award in 2012 from IEEE SMC Society, the Outstanding Electrical Engineering Professor Award from the Chinese Institute of Electrical Engineering in 2014, Outstanding Industry Contribution Award from Taiwan Systems Association in 2016, the best paper award in the International Journal of Fuzzy Systems in 2017, and many best paper awards from many international conferences technically supported by IEEE. He is the advisor, IEEE SMC student branch chapter at National Chung Hsing University; this chapter was the recipient of certificate of appreciation from IEEE SMCS in 2009. He has served as the associate editors of International Journal of Fuzzy Systems, and IEEE Transactions on Systems, Man and Cybernetics: Systems, IEEE Transactions on Industry Informatics, and International Journal of Electrical Engineering. Recently, he has served as the Editor-in-Chief of a new international robotics journal called “iRobotics”. His current interests include advanced nonlinear control methods, deep model predictive control, fuzzy control, neural-network control, advanced mobile robotics, intelligent service robotics, intelligent mechatronics, intelligent learning control methods with their applications to industrial processes and intelligent machinery.

Chien-Cheng Yu received the B.S. and M.S. degrees from Feng Chia University and National Taiwan Normal University, respectively. Now, he works as an Associate Professor with the Department of Electronic Engineering at Hsiuping University of Science and Technology, Taichung City, Taiwan. He is currently working toward the Ph.D. degree at National Chung Hsing University, Taichung City, Taiwan. His current research interests include memory architecture and circuit design, nonlinear control, and intelligent control.

Shih-Min Hsieh received the B.S. degree in electrical engineering from Feng Chia University, and the M.S. and Ph.D. degrees from the Department of Electrical Engineering, National Chung Hsing University, Taichung City, Taiwan. Now, he works as an Assistant Professor with the Department of Electronic Engineering at National Chin-Yi University of Technology, Taichung City, Taiwan. His current research interests include robotics, nonlinear control,

Feng-Chun Tai received the B.S., M.S. and Ph.D. degrees in Department of Electrical Engineering from National Chung Hsing University, Taichung, Taiwan, ROC. in 2007, 2010 and 2018, respectively. His current research interests include mobile robots, intelligent control, navigation system and their applications to industrial processes and machines.

Frontier Based Robot Exploration with Rapidly-Exploring Random Tree

Cheng-Yan Wu, Huei-Yung Lin, and Van Luan Tran

***Abstract*—In recent years, robots have been quickly integrated into people's daily lives. To allow the robots to navigate autonomously, accurate maps have to be provided. Therefore, it is important to make robots to obtain maps automatically and improve the efficiency of autonomous exploration. In this paper, we propose a method based on the rapidly-exploring random tree (RRT) and frontier 2D-SLAM exploration techniques. The proposed system is divided into three parts. First, we construct an initial map with laser range data, and use RRT and frontier detector to identify the frontier points of the initial map. The frontier points are then filtered and clustered to reduce the total number and the computation load. Finally, the score of each frontier point is calculated and the mobile robot is directed to the unknown areas until the map is constructed. In the experiments, the performance is evaluated in various synthetic scenes and real indoor environments. The results show that our system is able to successfully complete the autonomous exploration task in a reasonable time.**

***Index Terms*—** SLAM, autonomous exploration, rapidly-exploring random tree.

I. Introduction

MOBILE robots can move around in the real world environments. They are usually controlled by software and use various sensors to identify their surroundings. This allows them to navigate autonomously, and is probably able to replace human beings in some environments to execute specific tasks. One important prerequisite for mobile robots to perform various jobs is to explore the unknown environments successfully. This requires efficient exploration strategies for complete coverage and map construction. Using the sensor information acquired in the environment, path planning and region exploration can be carried out with minimum cost and time while ensuring the integrity of the map building.

This paper presents an approach which uses an RRT exploration strategy and a frontier-based image segmentation technique to find the frontiers between the explored region and unknown space. After filtering and scoring, the mobile robot is guided to the frontier point with the highest score to retrieve more map information. RRT is a path planning algorithm which can add nodes to the tree by sampling space using randomly generated points [1]. Based on RRT we can extend the search to unknown areas, and the probability of the extended space is proportional to the size of the Voronoi region [2]. Since the larger Voronoi region is closer to the unknown, the RRT algorithm tends to find the unexplored region and is suitable for the robot exploration task.

In previous works, Yanauchi proposed an algorithm based on frontier exploration [3]. In the method, he defined the line between the known and unknown regions as the frontier, generated candidate frontier points for environmental exploration, and comprehensively evaluated the exploration cost and associated information of each frontier point. According to the score, the optimal frontier point for the next step was determined. It was then used to direct the robot to the optimal frontier point, and increase the known area until the entire space is fully explored.

General frontier detection algorithms process the entire map data. This can be a time consuming process and slows down the exploration. Keidar *et al.* [7] proposed a wavefront frontier detector (WFD) based on a graph search which only detects the frontiers of known areas in the map, and a fast frontier detector (FFD) based on processing only the new laser readings data. Both algorithms do not process the entire map data and provide more efficient solution for uninterrupted exploration.

Bourgault *et al.* presented a different exploration algorithm using an information-based strategy [8]. They consider the localization and mapping problems concurrently with exploration. This reduces the uncertainty of the robot pose estimate and thus helps to maximize the map information. In order to calculate the robot's motion loss, moving distance and other issues, some researchers have introduced the concept of next best view (NBV) into the autonomous exploration of mobile robots [9]. The next best exploration point of a mobile robot is selected according to the corresponding evaluation.

This paper uses ROS (robot operating system) as a framework. The exploration system consists of SLAM, path planning, and exploring strategy [10]. The SLAM task is carried out by the Gmapping algorithm [11]. Gmapping adopts a highly efficient Rao-Blackwellized particle filer which uses grid based mapping with multiple particles. Navigation Stack provided by the ROS framework takes the information from odometry, sensors and goal poses, and outputs the safe velocity commands to the mobile robot. It enables the mobile robot to reach the

The support of this work in part by the Ministry of Science and Technology of Taiwan under Grant MOST 106-2221-E-194-004 and the Advanced Institute of Manufacturing with High-tech Innovations (AIM-HI) from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan is gratefully acknowledged.

Cheng-Yan Wuq Huei-Yung Lin are with the Department of Electrical Engineering, Advanced Institute of Manufacturing with High-Tech Innovation, National Chung Cheng University, Chiayi 621, Taiwan.

(Corresponding author Huei-Yung Lin, email: lin@ee.ccu.edu.tw)

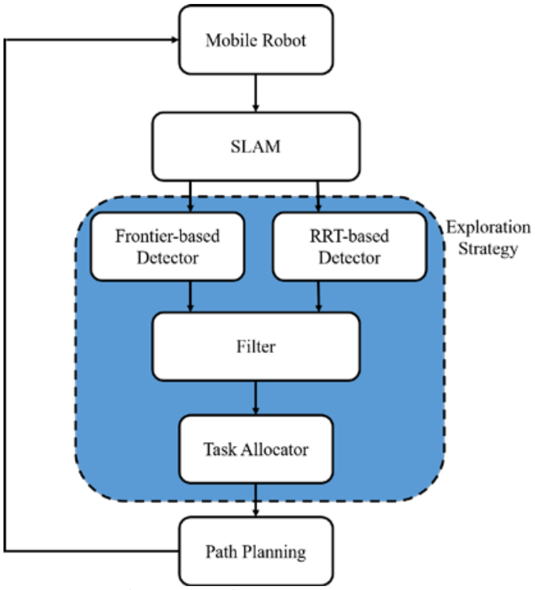


Fig. 1. The system flowchart of the proposed technique. Our exploration strategy consists of two detectors, a filter and a task allocator.

exploration goals and avoid obstacles. Global planner based on Dijkstra's algorithm [12] plans a path from the mobile robot's position to the exploration goal. Local planner is a reactive algorithm which uses the dynamic windows approach [13]. Both techniques are adopted in this work for autonomous exploration of a mobile robot.

II. EXPLORATION STRATERY

The system flowchart of the proposed technique is shown in Fig. 1. SLAM and path planning used in this work are introduced in Section I. First, an initial map is built based on the sensor information, and RRT and frontier based algorithms are used to search the frontier points of the initial map. Next, we filter the frontier points to reduce the total number of points and computation, and cluster them with the mean shift algorithm [14]. Finally, the score of each frontier point is calculated and the mobile robot is guided to the frontier point of highest score until the environment is completely explored.

A. *RRT-based Detector*

The RRT-based detector is used to search the frontier points around the mobile robot. If a point lies in an unknown region of the map and the growing RRT tree touches it, we will consider the point a frontier point. The detector is similar to the RRT algorithm, the tree starts from an initial point x_{init}. At each iteration, a random point x_{rand} is generated in the space. The node of an RRT tree is selected as $x_{nearest}$, which is nearest to x_{rand}. $x_{nearest}$ extends a distance from x_{rand} to generate x_{new}. Then, it checks if x_{new} lies in an unknown region of the map, or if there is any unknown region between x_{new} and $x_{nearest}$. If one of the above two conditions is true, x_{new} is considered as a frontier point and sent to the filter. The RRT tree is reset and other nodes are deleted.

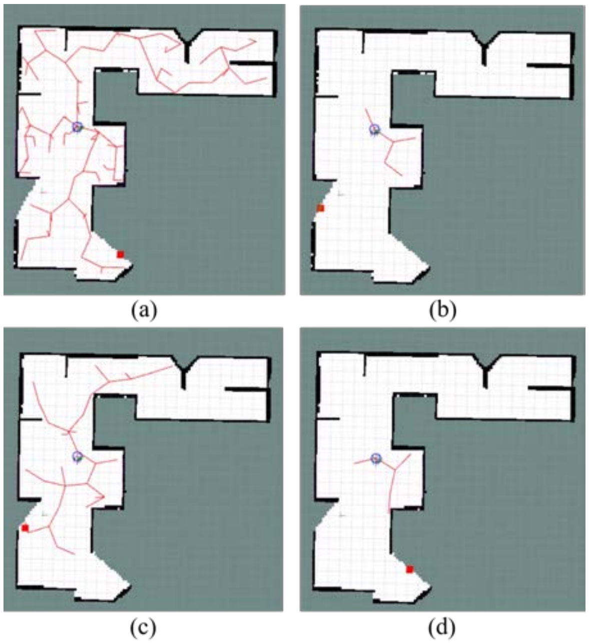


Fig. 2. RRT-based detector

The next iteration of the RRT tree starts from the current position of the mobile robot. That is, x_{init} is equal to $x_{current}$. If neither condition is true, it indicates that there is no obstacle at x_{new} and no obstacle between x_{new} and $x_{nearest}$. Then x_{new} is added as a new node to the RRT tree and is wired to $x_{nearest}$. It will return to generate a new x_{rand}, and the RRT tree continues to grow. The RRT tree is shown in Fig. 2.

B. *Frontier-based Detector*

The frontier-based exploration algorithm is currently the most widely used method for autonomous exploration. It is used to find the frontier points of the open space. When the mobile robot moves and constructs a new map, the frontier-based detector can instantly identify the frontier points on the map. Our frontier-based detector uses image segmentation techniques to extract the boundaries between known and unknown regions of the map. The extraction process is illustrated in Fig. 2.

We first construct the initial map from SLAM and convert the occupancy grid map to a grayscale map as shown in Fig. 2(a). The map is binarized to observe the distribution of obstacles as illustrated in Fig. 2(b). Then, we perform the dilation and NOT operations to obtain an image (see Fig. 2(c)), and apply the edge detection on the initial map to obtain an image including the obstacles and the boundaries between known and unknown regions of the map. Fig. 2(d) shows the edge detection result. An AND operation is carried out on Fig. 2(c) and Fig. 2(d) to eliminate the obstacles of the map. Consequently, we can derive an image only containing frontiers as shown in Fig. 2(e). The midpoints of the edges are finally selected as the frontier points (see Fig. 2(f)). The smaller length frontiers can be filtered out by adjusting the degree of obstacle dilation.

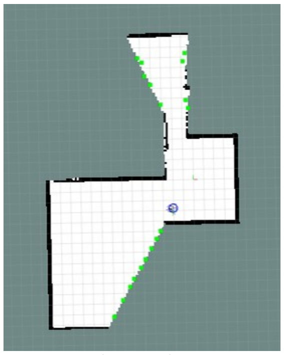
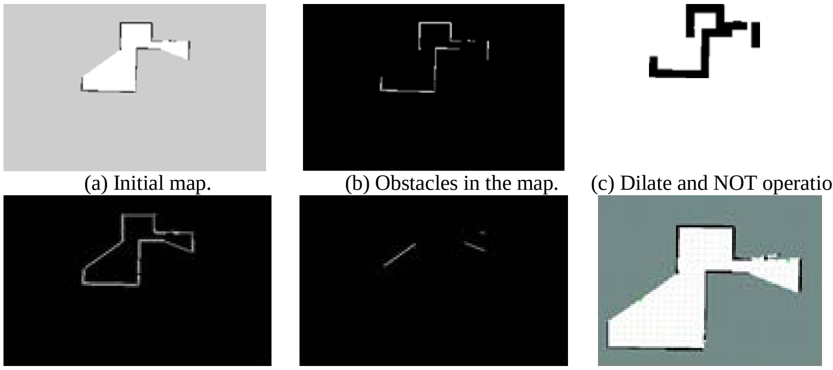


Fig. 3. The filtered frontier points



(a) Initial map.
(b) Obstacles in the map.
(c) Dilate and NOT operations.
(d) Edge detection.
(e) Frontiers of the map.
(f) The frontier points.

Fig. 4. The process of the frontier-based detector to extract the frontier point

The purpose of using two different detectors in this work is to compensate the disadvantages of the two methods. RRT will re-grow at the current position of the mobile robot when it finds the frontier point near-by. However, the mobile robot might miss the unexplored corners in some situations. Thus, we combine a frontier-based detector to search the entire space instantly, and this allows the mobile robot to explore more efficiently.

C. *Filter*

The filter receives all detected frontier points from the RRT and frontier based detectors, and there might be many points very close to each other. If all these points are sent to the task allocator to calculate the scores, it will take too much computation resource. Thus, we use mean shift algorithm to cluster these frontier points, and eliminate invalid and old frontier points to reduce the computation. This approach shifts each point in the direction of increasing probability. The points move iteratively to a higher density region until it converges. Fig. 3 shows the filtered frontier points.

D. *Task Allocator*

The task allocator receives the points processed from the filter and selects the point with the highest score among all the frontier points to assign to the mobile robot to perform the exploration task. Given a frontier point p_f and the current mobile robot position p_r, the score of each frontier point is calculated by

$$\begin{gathered} Score(p_r, p_f) = c_1 \cdot S_{info} - c_2 \cdot d_E \text{ where } c_1,\ c_2 \in \mathbb{R} \\ \mathrm{p}_f = \{(\mathrm{x};\ \mathrm{y}),\ \mathrm{x} \in \mathbb{R}, y \in \mathbb{R}\} \\ \mathrm{S}_{\mathrm{info}} = \pi r^2 \cap \mathrm{Area}_{\mathrm{unknown}} \\ \mathrm{d}_{\mathrm{E}} = \left\| p_r - p_f \right\|_2 \end{gathered} \tag{1}$$

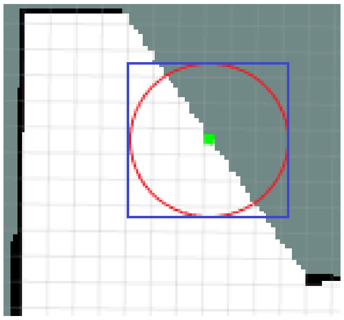


Fig. 5. Calculating the information gain of a frontier point

where the information gain S_{info} is the amount of information that can be obtained at the frontier point. The value S_{info} is derived by calculating the area of the unknown region within the radius r of the frontier point. It calculates the size of the unknown area in the blue square that are located in the red circle as shown in Fig. 5. d_E is the Euclidean distance between the current position of the mobile robot p_r and the frontier point p_f . We take d_E as the expected cost of moving the mobile robot to the frontier point p_f . However, the robot needs to consider other factors in the environment, and it is difficult to take a straight line to reach the frontier point. Thus, we multiply d_E by a constant c_2 to represent the cost of the mobile robot navigating to the frontier point.

In order to reduce the mobile robot's repeated exploration in two regions, we multiply the frontier point within the mobile robot radius d_r by a constant c_1. The information gain of the frontier point in the range will be enlarged, and this allows the mobile robot to be biased toward exploring the nearby areas.

$$\begin{cases} 1 & \text{if } \mathrm{d}_{\mathrm{E}} > \mathrm{d}_{\mathrm{r}} \\ 2 & \text{if } \mathrm{d}_{\mathrm{E}} \leq \mathrm{d}_r \\ 2 & \text{if } \left\| p_m - p_f \right\|_2 \leq d_r \end{cases} \tag{2}$$

where p_m is the frontier point that the mobile robot is navigating to, so that the information gain of the frontier point near the task point is increased. When the nearby area is completely explored, it will go to the next area, effectively reducing the possibility of the mobile robot repeatedly exploring.

III. Experiments

The proposed RRT and frontier based exploration system is compared with the greedy frontier-based algorithm [15] and RRTs algorithm [16]. The information gain radius is 1.0 m and we let d_r =3.0 m. Each environment is successfully tested for 20 times, and we select 10 normal working results. The moving speed of the mobile robot is 0.3 m per second.

The mobile robot platform used in the experiments is a Pioneer P3-DX. A laptop computer with Intel Core i7-7700 CPU @3.6 GHz and 8 GB RAM is used as the computing platform. The operating system is Ubuntu 16.04 with ROS Kinetic. A SICK LMS 100 laser range scanner with the working range of 0.5 m ~20 m, -450° ~ 255°, 50 Hz, and an

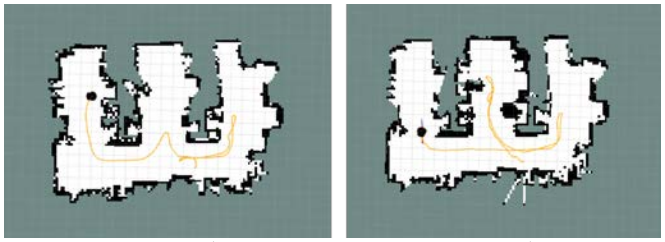


Fig. 6. RRT-based and frontier-based exploration strategies for the laboratory environment. The left map takes less exploration time.

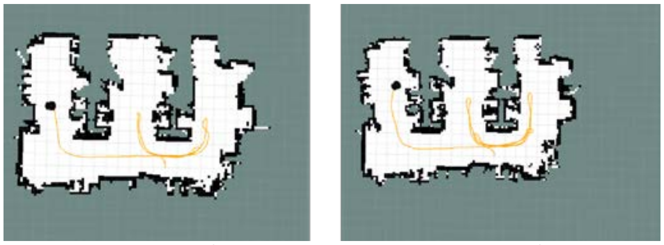


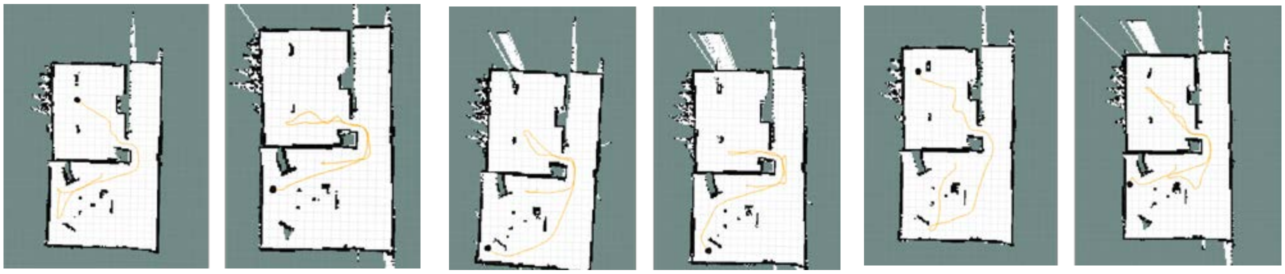
Fig. 7. RRTs algorithm for the laboratory environment. The left map takes less exploration time.

Astra RGB-D camera with the range of 0.6 m ~ 8 m, horizontal 60°, vertical 49.5° are used for data acquisition. In addition, an embedded computer with Nvidia Jetson TX2 is used for onboard processing.

The laboratory environment for our real world experiments and the results are shown in Figs. 6 and 7. Table I shows our exploration approach compared with the RRTs algorithm. It reduces 7.3% exploration time and 18.5% navigation path length. The second experiment is carried out in a classroom corridor environment. The results are shown in Fig. 8. Table II shows our exploration technique compared with the RRTs and greedy frontier-based algorithms. Compared to RRTs, our algorithm reduces 32.6% in the exploration time and 36.1% in the navigation path length. As for the comparison with the greedy frontier-based algorithm, ours reduces 22.3% in the exploration time and 24.1% in the navigation path length.

TABLE I
THE EXPLORATION TIME (S) AND EXPLORATION PATH LENGTH (M) IN THE REAL ENVIRONMENT.

	Exploration Time (s)		Exploration Length (m)	
Algorithm	Ours	RRTs	Ours	RRTs
1	86.20	127.90	19.08	29.59
2	116.20	103.68	18.58	26.45
3	70.20	72.82	13.40	15.59
4	113.81	96.40	21.74	21.40
5	90.01	90.20	16.98	19.08
6	112.80	101.60	24.06	26.09
7	74.21	112.20	16.85	27.79
8	77.01	95.41	17.47	19.72
9	93.00	90.60	20.65	21.03
10	101.80	117.40	22.87	28.18
Average	93.52	100.82	19.16	23.49



(a) RRT-based and frontier-based exploration strategies for the classroom and corridor environment. The left map takes exploration time.

(b) RRTs algorithm for the classroom and corridor environment. The left map takes less exploration time.

(c) Greedy frontier-based algorithm for the classroom and corridor environment.

Fig. 5. The results of the classroom and corridor environment

TABLE II
THE EXPLORATION TIME (S) AND EXPLORATION PATH LENGTH (M) IN THE CLASSROOM AND CORRIDOR ENVIRONMENT.

	Exploration Time (s)			Exploration Length (m)		
Algorithm	Ours	RRTs	Greedy frontier-based	Ours	RRTs	Greedy frontier-based
1	105.75	95.20	124.78	22.47	23.50	24.13
2	87.13	127.61	104.01	19.71	24.90	22.77
3	74.94	126.80	104.11	17.05	30.72	25.22
4	93.51	150.00	92.33	21.54	38.03	21.88
5	86.54	146.60	119.84	16.96	36.97	28.59
6	68.61	141.06	88.11	18.57	32.94	21.68
7	63.06	111.60	104.18	16.09	24.02	23.69
8	69.57	105.21	107.76	15.03	24.93	25.43
9	96.68	101.00	120.56	21.77	24.04	27.99
10	83.50	124.60	100.78	16.48	30.58	23.33
Average	82.93	122.97	106.65	18.57	29.06	24.47

IV. Conclusion

In this paper, we propose an autonomous exploration system based on RRT and frontier. It allows the mobile robot to explore independently in an unknown indoor environment. Since RRT is a random search algorithm, when the space contains many obstacles or the robot is in narrow space, the algorithm will converge slower with inefficient results. When exploring a new unknown areas, it cannot immediately grow toward new known regions, causing the slow search of frontier points. The mobile robot will then go to other frontier points, which may cause repeated exploration. In order to solve the above problems, this paper has added a frontier-based exploration strategy that enables instant search for the frontier points. Our system has preliminary results in the exploration of basic functions with experiments in simulation and real world environments. The results have demonstrated the feasibility of the proposed technique.

Acknowledgment

The support of this work in part by the Ministry of Science and Technology of Taiwan under Grant MOST 106-2221-E-194-004 and the Advanced Institute of Manufacturing with High-tech Innovations (AIM-HI) from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan is gratefully acknowledged.

References

[1] S. M. Lavalle, “Rapidly-exploring random trees: A new tool for path planning,” Tech. Rep., 1998.

[2] X. Z. Peng, H. Y. Lin, and J. M. Dai, “Path Planning and Obstacle Avoidance for Vision Guided Quadrotor UAV Navigation,” *The 12th IEEE International Conference on Control & Automation (ICCA 2016)*, pp. 984-989, Kathmandu, Nepal, June 1-3, 2016.

[3] B. Yamauchi, “A frontier-based approach for autonomous exploration,” *1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, July 1997, pp. 146-151.

[4] G. Oriolo, M. Vendittelli, L. Freda, and G. Troso, “The srt method: randomized strategies for exploration,” *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA 04. 2004*, vol. 5, April 2004, pp. 4688-4694.

[5] H. El-Hussieny, S. F. M. Assal, and M. Abdellatif, “Improved backtracking algorithm for efficient sensor-based random tree exploration,” *2013 Fifth International Conference on Computational Intelligence, Communication Systems and Networks*, June 2013, pp. 1924.

[6] T.-B. Kwon and J.-B. Song, “Thinning-based topological exploration using position probability of topological nodes,” *2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*, May 2006, pp. 797-802.

[7] M. Keidar and G. A. Kaminka, “Robot exploration with fast frontier detection: theory and experiments,” *11th International Conference on Autonomous Agents and Multiagent Systems, 2012*, pp. 113-120.

[8] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, “*Information based adaptive robotic exploration,” in IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, Sept 2002, pp. 540-545.

[9] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova and R. Siegwart, “Receding Horizon “Next-Best-View” *Planner for 3D Exploration,” 2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, 2016, pp. 1462-1468.

[10] C. H. Do and H. Y. Lin, “Simultaneous Localization and Mapping with Neuro-Fuzzy Assisted Extended Kalman Filtering,” *2017 IEEE/SICE International Symposium on System Integration (SII 2017)*, Taipei, Taiwan, Dec 11-14, 2017.

[11] G. Grisetti, C. Stachniss, and W. Burgard, “Improved techniques for grid mapping with rao-blackwellized particle filters,” *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 3446, Feb 2007.

[12] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische mathematik*, vol. 1, no. 1, pp. 269271, 1959.

[13] D. Fox, W. Burgard, and S. Thrun, “The dynamic window approach to collision avoidance,” *IEEE Robotics Automation Magazine*, vol. 4, no. 1, pp. 2333, March 1997.

[14] D. Comaniciu and P. Meer, “Mean shift: a robust approach toward feature space analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603619, May 2002.

[15] J. Hrner, “Map-merging for multi-robot system,” Prague, 2016.

[16] H. Umari and S. Mukhopadhyay, “Autonomous robotic exploration based on multiple rapidly-exploring randomized trees,” *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 1396-1402.

[17] A. Kim and R. M. Eustice, “Active visual SLAM for robotic area coverage: Theory and experiment,” *The International Journal of Robotics Research*, 34.4-5 (2015): 457-475.

[18] “Gazebo,” http://gazebosim.org/.

Huei-Yung Lin received his BS degree in Applied Mathematics from National Chiao Tung University, Taiwan, and his MS and PhD degrees in electrical and computer engineering from the State University of New York at Stony Brook. In 2002 he joined the Department of Electrical Engineering, National Chung Cheng University, Taiwan, and currently is a full professor. His current research interests include computer vision, robotics, pattern recognition, and image processing. He serves as an organizing committee member of several robotics related IEEE international conference. He is also a founding member and deputy secretary general of Taiwan Society of Robotics.

Potential-Field-Based Distributed Formation Control Using Consensus Algorithms and PSO-RGA for Small-Scale Unmanned Multi-Helicopters

Ching-Chih Tsai, *Fellow, IEEE,* Ching-Fu Hsu, Zen-Chung Wang, and Feng-Chun Tai

Abstract-**This paper presents a distributed formation control system with a trajectory generation method using consensus algorithm, potential field (PF), particle swarm optimization (PSO) and real-coded genetic algorithm (RGA) for a group of small-scale unmanned helicopters (SSUHs) whose low-level trajectory tracking controllers have been constructed by using the fuzzy basis function networks (FBFN). The distributed formation trajectory generator using a virtual leader-follower structure is proposed by an improved 3D potential function with consensus algorithm, and the key parameters of the generator are optimally searched by using the combination of PSO and RGA algorithms, called PSO-RGA algorithm. The performance and merits of the proposed method is exemplified by conducting four simulations on a group of four small-scale unmanned helicopters cooperatively flying over environments with complicated terrain. These simulation results indicate that all the following helicopters not only exactly achieve consensus tracking and track their positions and headings given by the virtual leader, but also attain good abilities to achieve collision and obstacle avoidance among helicopters and obstacle avoidance between the helicopters and complex terrain.**

***Index Terms*—Consensus, formation control potential field, real-coded genetic algorithm, small-scale unmanned helicopter.**

I. Introduction

RECENTLY, multi-robot systems have been extensively used not only for industrial applications, but also for many other applications in military, aerospace, human services, education and our daily life. On basis of interaction types occurring in typical applications, mobile multi-robot systems can be classified into five categories: collective, cooperative, collaborative, coordinate and adversary [1]. Cooperative formation control problems using the combinations of collective and cooperative techniques have received much attention in control community [2-4]. Such cooperative formation control technologies have gained many applications to actual vehicles, robots and agents, including unmanned aerial vehicles (UAVs), artificial satellites, and autonomous mobile robots, automatic guided vehicles (AGVs) in manufacturing, and distributed sensor networks [3-4]. Moreover, since distributed intelligence has been regarded as a powerful paradigm for multi-robot systems [1], distributed control has been considered as a useful and practical method for multi-robot control systems due to the fact that each distributed controller on each robot only obtains required information from itself and its neighboring robots.

Cooperative formation control problems have been widely addressed by many researchers [5-8]. Generally speaking, formation controllers for unmanned helicopters are designed to not only achieve desired formation without collisions between vehicles, but also maintain the achieved formation while tracking a desired path as a team. Most of these control approaches have been designed using consensus algorithms since such control methods take the advantages of network flexibility, but few have been done using other approaches, such as 3D potential functions [9], hybrid modeling and control [10-12], virtual structures [13], behavior schemata [14], leader-follower structure [15], local control architectures[15-16], graph theory [17-20] and so on. Although those methods in [9-10] have been shown to achieve satisfactory formation control performance for unmanned multi-helicopters in free space, the cooperative formation control problem in terrain-dependent environments remains to be done for small-scale unmanned multi-helicopters with distributed control approaches.

Small-scale unmanned helicopters in military applications have been shown suitable for a variety of military tasks in complicated terrain and environments; those tasks include enemy reconnaissance, route survey, electromagnetic interference or attack specific targets and so on. In spite of their limited payload and endurance and other factors, such helicopters show their strengths and merits in contact with the enemy in the junction area of operations. Before the war intelligence grasps and deploys effective fighting force, these SSUHs can increase the contact with the enemy, thereby reducing the possibility of early injury of soldiers to perform reconnaissance mission. Except military applications, SSUHs can be applicable to many civil applications, such as bridge diagnosis, detailed area mapping, close-up inspection, environmental monitoring and protection, agriculture, mineral exploration and exploitation and so on.

To find the optimal control signals for multi-UAV formation, many researchers have proposed methods using PSO and GA. For example, Duan et al. [21] proposed a hybrid PSO-GA algorithm to find the best control signals for multi-UAV formation reconfiguration. Similar to the hybrid PSO-GA algorithm, a variety of the combinations of PSO and RGA have been investigated by many researchers

Ching-Chih Tsai, Ching-Fu Hsu, Zen-Chung Wang, and Feng-Chun Tai are with the Department of Electrical Engineering, National Chung Hsing University, Taichung, 40227, Taiwan.
(Corresponding author: Ching-Chih Tsai, email: cctsai@nchu.edu.tw) (email: hchingfu@gmail.com, zenchung1125@gmail.com , fctai@nchu.edu.tw)

The authors gratefully acknowledge financial support from Ministry of Science and Technology (MOST), Taiwan, ROC, under contract MOST 107-2221-E-005 -073-MY2.

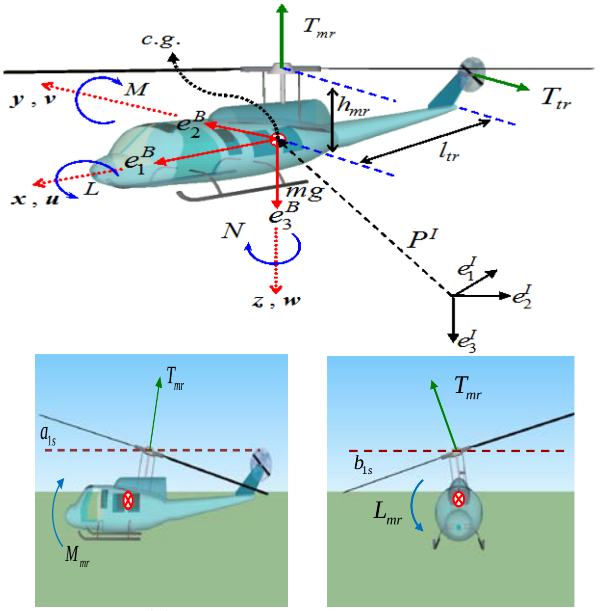


Fig. 1. Illustration of forces and moments acting on a small-scale unmanned helicopter (SSUH).

[22-25]. Two examples in [24-25] were used to show how the combined PSO-RGA algorithms were effective in not only solving for inverse kinematics of seven-degrees-of freedom manipulators, but also designing optimal PID controllers. However, this PSO-RGA algorithm in series has not yet been applied to tune the PF-based formation controller for a group of small-scale unmanned helicopters.

In recent years, the concept of the fuzzy basis function networks (FBFN) has become increasingly important. It has been known that the FBFN can approximate any continuous function over a compact set and have high accuracy and fast learning ability. Moreover, FBFNs have also been proved to excellently approximate time-varying nonlinear functions or nonlinear dynamics. This merit can be easily applied to controller design for such systems. For example, Tsai et al. [26] proposed an intelligent adaptive control method along with FBFN for trajectory tracking of an autonomous small-scale helicopter. Motivated by [26], the low-level FBFN-based motion controller would be helpful and powerful in tracking formation trajectories generated for a group of SSUHs flying together in formation.

The objectives of the paper are to establish a distributed formation control system with a novel trajectory generation method for a group of SSUHs cooperatively flying over complicated terrain using consensus algorithm, potential field (PF) and PSO-RGA algorithm, and to verify the effectiveness and merit of the proposed control method via simulations. The merit of the distributed formation controller for each SSUH is that each controller only takes required information from itself and its neighboring SSUHs via the communication topology. By comparing to the state of the art in the field of formation control, the paper is novel in proposing an effective and practical distributed formation control law for a group of SSUHs in a complicated environment with static and dynamic obstacles.

The rest of this paper is constructed by follows. Section II briefly describes modeling and low-level FBFN-based control of single small-scale unmanned helicopter. In Section III, the distributed formation control system is presented and then its formation trajectory generation method is proposed by using the improved 3D potential function and consensus algorithm. Section IV investigates the asymptotical stability and consensus analysis of the proposed formation trajectory generation method. Section IV uses the PSO-RGA algorithms to offline find the best controller parameters optimally. Simulations are done in Section VI to demonstrate the performance and merit of the proposed system. Section VII concludes the paper.

II. Modeling, Low-level Control and Problem Statement

The overall dynamics of a small scale helicopter is considered as a rigid-body dynamics actuated by thrust force and wrench moments resulted from other dynamics which include rotor wing aerodynamics, rotor flapping dynamics, engine dynamics and actuator dynamics in [27]. The controller design must be entirely based on the rigid-body dynamics without other state variables since they can be hardly measured online. Therefore, the rigid-body model of helicopter is briefly recalled here for the control design purpose, and then the low-level trajectory tracking controller using the FBFN on-line approximator is briefly described, in order to test the planned global paths by the proposed method.

A. *Modeling a SSUH*

Fig.1 illustrates forces and moments exerting on an autonomous small scale helicopter. With the symbols notations in Fig.1, the equations of motion of a helicopter are given by the Newton-Euler equations and summarized as follows [26-27].

$$\dot{P}^{\mathrm{I}} = v^{\mathrm{I}} \tag{1}$$

$$\dot{v}^{\mathrm{I}} = ge_3^{\mathrm{I}} + \frac{1}{m}\Re(\Theta)u_T^{\mathrm{B}} + \frac{1}{m}\Re(\Theta)\boldsymbol{K}u_M^{\mathrm{B}} \tag{2}$$

$$\dot{\Re}(\Theta) = \Re(\Theta)sk(\omega^{\mathrm{B}}) \tag{3}$$

$$I_m\dot{\omega}^{\mathrm{B}} = -\omega^{\mathrm{B}} \times I_m\omega^{\mathrm{B}} - Q_{tr}e_2^{\mathrm{B}} - Q_{mr}e_3^{\mathrm{B}} + \boldsymbol{J}u_M^{\mathrm{B}} \tag{4}$$

where $P^{\mathrm{I}} = \left[x\ y\ z\right]^T$ and $v^{\mathrm{I}} = \left[v_x\ \ v_y\ \ v_z\right]^T$ are position and velocity of the helicopter center of gravity (c.o.g.) with respect to the inertial frame in North-East-Down (NED) orientation, e_j^{I} and e_j^{B} are denoted the j^{th} unit vectors of the inertial frame and body frame, $m \in R$ and $I_m \in R^{3\times3}$ respectively denote the total mass and the moment inertial matrix of the robotic helicopter. The Euler angle vector $\Theta^{\mathrm{B}} = [\phi\ \theta\ \psi]^T$ and the helicopter angular rate vector $\omega^{\mathrm{B}} = \left[\omega_x\ \ \omega_y\ \ \omega_z\right]^T$ defined in the roll-pitch-yaw sequence are with respect to its body frame. The four independent inputs to this model are lift force control denoted by $u_T^{\mathrm{B}} = -T_{mr} e_3^{\mathrm{B}}$, and three directional torque controls denoted by $u_M^{\mathrm{B}} = \left[L_{mr} \ M_{mr} \ N_{tr} \right]^T \in R^3$ where T_{mr} is represented a thrust force and L_{mr}, M_{mr}, N_{tr} are defined the three moments. Both u_T^{B} and u_M^{B} are applied along the body frame. Q_{mr} and Q_{tr} respectively represent the anti-torques on the main rotor and tail rotor. Both matrices $\boldsymbol{J}$ and $\boldsymbol{K}$ are related to the helicopter mechanical characteristics and expressed by

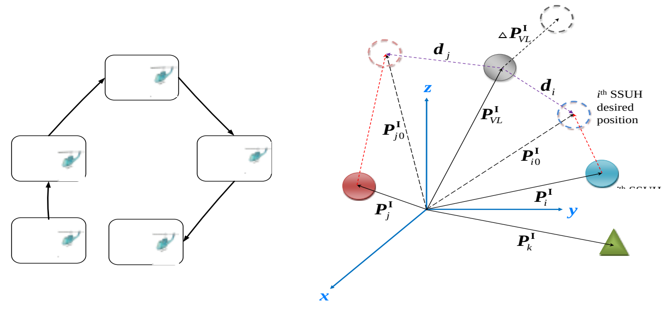
$$\boldsymbol{J} = \begin{bmatrix} 1+\dfrac{k_b}{T_{mr}h_{mr}} & 0 & 0 \\ 0 & 1+\dfrac{k_b}{T_{mr}h_{mr}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{K} = \begin{bmatrix} 0 & \dfrac{-1}{h_{mr}} & 0 \\ \dfrac{1}{h_{mr}} & 0 & \dfrac{-1}{l_{tr}h_{mr}} \\ 0 & 0 & 0 \end{bmatrix} \quad (5)$$

Furthermore, $\Re(\Theta)$ is the helicopter's rotation matrix from the body axes to the inertial axes and $sk(\omega^{\mathrm{B}})$ means the skew-symmetric matrix of the body angular rate.

The four independent inputs to this model are lift force where h_{mr} denotes the height of rotor head above the helicopter centre of gravity and a constant stiffness coefficient k_b is used to represent the restraint torque in the blade attachment to the rotor head and the anti-torques on the rotors. Because the lift force T_{mr} ensure greater than zero, the matrix $\boldsymbol{J}$ is bound to be positive definite. Besides, l_{tr} is the distance from the c.o.g. to tail rotor hub.

B. *Low-Level Trajectory Tracking Control*

This subsection is devoted to briefly describing the low-level trajectory tracking controller of the autonomous helicopter using FBFN in [26]. For the trajectory tracking, the three positions and the heading of the helicopter are chosen as the desired outputs, and the desired trajectories are given by the position vector, $P_d^{\mathrm{I}}(t)$, the heading angle, $\psi_d(t)$ and their time derivatives ($\dot{P}_d^{\mathrm{I}}$, $\ddot{P}_d^{\mathrm{I}}$, $\dddot{P}_d^{\mathrm{I}}$, $\dot{\psi}_d$, $\ddot{\psi}_d$). The control objective is to find the control laws for u_T^{B} and u_M^{B} such that all the output errors (δ_i, i =1, ... ,4 and ε_j, j =1, 2) can converge towards zero for any initial condition. For tracking any trajectories of an autonomous helicopter, the proposed method is based on the full rigid body dynamics (1)-(4) where the coupling forces in (2) are not neglected. All control inputs of the thrust (u_T^{B}) and moments (u_M^{B}) are explicit early in the second error dynamic equation ($\dot{\delta}_2$). To successfully complete the trajectory tracking controller in [26], backstepping control design and two techniques of the dynamic extension are adopted for the thrust control and the moment controls respectively, and the on-line FBFN approximator is employed to on-line approximate the small body forces and expressed as $\hat{\Delta} = \hat{W}^T \hat{S} \cong \Re(\Theta) K u_M^{\mathrm{B}} / m$



(a) (b)

Fig. 2. An example of communication topology in the helicopter formation control system: (a) the four follower SSUHs with the fifth SSUH as the virtual leader; (b) symbols used for formation flight.

where $\hat{S}$ is the estimate of the output vector of the hidden layer, and $\hat{W}$ denotes the estimate of the connective weight matrix W=$[w_{jk}]_{N\times 3}$ between the hidden layer and the output layer. In this paper, the FBFN is adopted as estimator in order to illustrate a coupling effect of unmanned helicopter. The node numbers of the input layer, the hidden layer, and the output layer are 8, N, and 3 respectively, where N is a positive integer. Both input and output layers are linear, and the activation function used by all nodes the hidden layer is a Gaussian function.

To accomplish the low-level motion control for each SSUH in [26], it is required to first find the so-called pseudo control signals ($\ddot{T}_{mr}$, $\dot{\omega}$) and then compute the actual control signals (u_T^{B}, u_M^{B}), thereby obtaining the actual force control u_T^{B} and the actual moment control u_M^{B} as below,

$$u_T^{\mathrm{B}} = \int_0^t \int_0^t \ddot{T}_{mr} d\tau \cdot d\tau \quad (6)$$

$$u_M^{\mathrm{B}} = J^{-1} \left[I_m \dot{\omega}^{\mathrm{B}} + \omega^{\mathrm{B}} \times I_m \omega^{\mathrm{B}} + Q_{tr} e_2^{\mathrm{B}} + Q_{mr} e_3^{\mathrm{B}} \right] \quad (7)$$

C. *Modeling a Multi-SSUH System*

Assume that the communication topology of n follower SSUHs is a directed subgraph G, and n SSUHs can be regarded as n nodes. The relevant weighted adjacency matrix of the subgraph $\boldsymbol{G}$ is denoted as $\boldsymbol{A} = [a_{ij}]$ and $a_{ij} \geq 0, \ \forall i, j \in \{1, 2, \ldots, n\}$. Moreover we assume $a_{ii} = 0$. The Laplacian matrix $\boldsymbol{L}$ of the directed subgraph $\boldsymbol{G}$ is defined as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$ where $\boldsymbol{D} = diag(d_1, d_2, \ldots, d_n)$ and $d_i = \sum_{j=1}^{n} a_{ij}$. Similarly, the overall multi-SSUH system under consideration consists of the n follower SSUHs and a virtual leader SSUH considered as the $(n+1)^{th}$ SSUH. The interconnection topology of the overall multi-SSUH system is also a directed graph $\bar{\boldsymbol{G}}$, and its Laplacian matrix $\bar{\boldsymbol{L}}$ is given by $\bar{\boldsymbol{L}} = \bar{\boldsymbol{D}} - \bar{\boldsymbol{A}}$. Fig. 2 illustrates a formation example of the four follower SSUHs with the fifth SSUH as the virtual leader where the least information is exchanged. In order to achieve this formation control objective, three assumptions about the communication topology are made in the following.

Assumption 1: The subgraph $\bar{\boldsymbol{G}}$ with its relevant Laplacian matrix $\bar{\boldsymbol{L}}$ is directed, connected and has a spanning tree.

Assumption 2: Every SSUH must be connected from the leader on the network, but all of the SSUHs are not necessarily directly connected from the leader.

Assumption 3: Movement of the leader must be independent from any SSUH.

Assumption 4: Every SSUH gets information regarding its position and heading and formation patterns; especially, $\boldsymbol{d}_i$ denotes the desired relative displacement vector between the i^{th} SSUH and leader, as shown in Fig. 2(b).

According to Assumption 1, the Laplacian matrix $\bar{\boldsymbol{L}} \in R^{(n+1)\times(n+1)}$ is symmetric and has only one zero eigenvalue and all other nonzero eigenvalues with positive real parts. On basis of Assumptions 1-3, all the entries in the last row of Laplacian matrix $\bar{\boldsymbol{L}} \in R^{(n+1)\times(n+1)}$ are zeros, and there exists a diagonal matrix $\boldsymbol{B} = diag\{a_{1(n+1)},\dots,a_{n(n+1)}\} \in R^{n\times n}$ where it has at least one positive diagonal entry such that the summation of both matrices, i.e., $\boldsymbol{L}$+$\boldsymbol{B}$, is invertible and has all the nonzero eigenvalues with positive real parts; therefore, there exist a symmetric and positive-definite matrix $\boldsymbol{S}$, and a symmetric and diagonal matrix $\boldsymbol{R}$ such that $\boldsymbol{R}(\boldsymbol{L}+\boldsymbol{B})+(\boldsymbol{L}+\boldsymbol{B})^T\boldsymbol{R}=\boldsymbol{S}$ where the superscript T denotes the transpose operation in vector and matrix operations. Moreover, matrix $\boldsymbol{R}$ can be found by $\boldsymbol{R}$=*diag*$\{r_1,\dots,r_n\}$, where $[1/r_1,\dots,1/r_n]^T$=$(\boldsymbol{L}$+$\boldsymbol{B})^{-1}[1,\dots,1]^T$ [28-29].

D. Problem Statement

The formation mission is to let n SSUHs follow their leader in formation as shown in Fig. 2. In achieving this mission, the distributed formation trajectory generation approach in this paper is stated as follows. Since the low-level FBFN-based motion controller of each SSUH has been designed and shown asymptotically stable, the aim of the proposed formation trajectory generation method is to generate stable formation trajectories for all the followers if the leading trajectory of the virtual leader is given. Worthy of mention is that the generated trajectories not only keep all the followers flying together in formation, but also accomplish collision and obstacle avoidance when the static and dynamic obstacle appear randomly in the flight routes. Note that the distributed formation control system is synthesized to meet Assumptions 1-4.

III. Formation Control Using Improved 3D Potential Field and Consensus Algorithm

In applying the potential field method, all the SSUHs are regarded a particle, and the environment is represented as a potential field which generates attractive or repulsive forces. In this section, a modified PF (MPF) method is proposed based on the well-known leader-follower structure and consensus concept. Fig. 3 depicts the block diagrams of the proposed MPF-based formation control system, where the terrain contour flight trajectory of the virtual leader is generated by our previous RGA-and-elastic-band trajectory planning method with a time-varying flight speed [30]. In the MPF method, the potential field of each SSUH is composed of four force components caused by the virtual leading force, mutual influencing force, collision-avoidance force, and obstacle avoidance force. Therefore, the total MPF force is the summation of the four forces. Unlike the conventional PF method, this modified PF scheme incorporates with the concept of consensus algorithm to address a more general distributed formation control problem. Below is the detailed description of each force in the proposed MPF method.

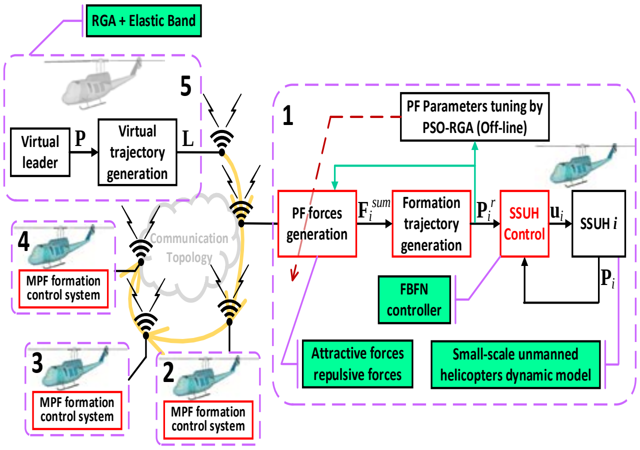

RGA + Elastic Band
5
Virtual leader
P
Virtual trajectory generation
L
Communication Topology
1
PF Parameters tuning by PSO-RGA (Off-line)
PF forces generation
$\mathbf{F}_i^{sum}$
Formation trajectory generation
$\mathbf{P}_i^r$
SSUH Control
$\mathbf{u}_i$
SSUH *i*
$\mathbf{P}_i$
FBFN controller
Attractive forces repulsive forces
Small-scale unmanned helicopters dynamic model
4
MPF formation control system
3
MPF formation control system
2
MPF formation control system

Fig.3. Block diagram of the proposed distributed MPF formation control system.

A. Virtual Leading Force

The virtual leader is the key reference point of each formation and dominates the formation movement. Depending on the underlying control system its trajectory can either be given as navigation points or as continuous trajectory. The virtual leading force of each follower SSUH is expressed by

$$\begin{aligned}\boldsymbol{F}_i^{VL} &= a_{i(n+1)}k_{VL}\times\left(\boldsymbol{P}_{VL}^{\mathrm{I}}-\boldsymbol{P}_i^{\mathrm{I}}-\left[\boldsymbol{P}_{VL}^{\mathrm{I}}-\boldsymbol{P}_{i0}^{\mathrm{I}|}\right]\right)\\ &= -a_{i(n+1)}k_{VL}\times\left(\boldsymbol{\mathcal{P}}_i^{\mathrm{I}}-\boldsymbol{P}_{VL}^{\mathrm{I}}\right) = -a_{i(n+1)}k_{VL}\times\left(\boldsymbol{\mathcal{P}}_i^{\mathrm{I}}-\boldsymbol{\mathcal{P}}_{n+1}^{\mathrm{I}}\right)\end{aligned} \tag{8}$$

where $\boldsymbol{P}_{VL}^{\mathrm{I}}$ stands for the position vector of the virtual leader; $\boldsymbol{P}_i^{\mathrm{I}}$ and $\boldsymbol{P}_{i0}^{\mathrm{I}}$ respectively represent the current and desired position vectors of SSUH i, as shown in Fig. 2(b), respectively; $\boldsymbol{\mathcal{P}}_i^{\mathrm{I}} = \boldsymbol{P}_i^{\mathrm{I}} - \mathcal{d}_i$, $\boldsymbol{P}_{i0}{}^{\mathrm{I}} = \boldsymbol{P}_{VL}^{\mathrm{I}} + \mathcal{d}_i$, $i=1,\dots,n+1$; k_{VL}>0 is the positive gain to be determined. Note that $\mathcal{d}_{n+1}=0$, namely that there is no relative displacement vector between the leader and itself such that $\boldsymbol{\mathcal{P}}_{n+1}^{\mathrm{I}} = \boldsymbol{P}_{VL}^{\mathrm{I}} + \mathcal{d}_{n+1} = \boldsymbol{P}_{VL}^{\mathrm{I}}$. Physically, this virtual leading force component guides each SSUH directly to its desired position relative to the virtual leader. If the velocity of the virtual leader is known for any follower SSUH which directly accesses the position and velocity information of the leader, the second virtual leading force vector for the whole multi-SSUH system is proposed by

$$\boldsymbol{F}^{VVL} = \dot{\mathcal{d}} + ((\boldsymbol{L}+\boldsymbol{B})^{-1}\otimes\boldsymbol{I}_3)(\boldsymbol{b}\otimes\dot{\boldsymbol{P}}_{VL}^{\mathrm{I}}) \tag{9}$$

where $\dot{\mathcal{d}} = [\dot{\mathcal{d}}_1,\cdots,\dot{\mathcal{d}}_n]^T$ and $\boldsymbol{b} = [a_{1(n+1)},\cdots,a_{n(n+1)}]^T$ is a nonzero vector which has at least one unity entry. From (9), it is easy to find the distributed consensus formation control law for the i^{th} SSUH in formation as follows;

$$\boldsymbol{F}_i^{VVL} = \dot{\mathscr{d}}_i + \sum_{j=1}^{n} \bar{l}_{ij} b_j \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}} \tag{10}$$

where the matrix $\boldsymbol{L}+\boldsymbol{B}$ is invertible and has its inversion denoted by $(\boldsymbol{L}+\boldsymbol{B})^{-1} = [\bar{l}_{ij}],\ i,j=1,\ldots,n$. Note that $\bar{l}_{ij}$ is not zero in the path of the communication topology; otherwise, $\bar{l}_{ij}$ is zero. On the other hand, the second term control law for the i^{th} SSUH in (10) can also be calculated by a recursive formula $\boldsymbol{F}_{2i}^{VVL} = \left(\sum_{j=1}^{n}(a_{ij}+b_i)\right)^{-1}\left(b_i \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}} + \sum_{j=1}^{n} a_{ij}\boldsymbol{F}_{2j}^{VVL}\right)$ such that $\boldsymbol{F}_i^{VVL} = \dot{\mathscr{d}}_i + \boldsymbol{F}_{2i}^{VVL},\ i=1,\ldots,n$.

B. *Mutual Influencing Force*

The contribution of another helicopter to the modified potential field is expressed by:

$$\begin{aligned}\boldsymbol{F}_{ij} &= a_{ij}k_{ij} \times \left(\boldsymbol{P}_j^{\mathrm{I}} - \boldsymbol{P}_i^{\mathrm{I}} - \left[\boldsymbol{P}_{j0}^{\mathrm{I}} - \boldsymbol{P}_{i0}^{\mathrm{I}}\right]\right)\\ &= a_{ij}k_{ij} \times \left(\boldsymbol{P}_j^{\mathrm{I}} - \boldsymbol{P}_i^{\mathrm{I}} - \left[(\boldsymbol{P}_{VL}^{\mathrm{I}} + \mathscr{d}_j) - (\boldsymbol{P}_{VL}^{\mathrm{I}} + \mathscr{d}_i)\right]\right) = -a_{ij}k_{ij} \times \left(\boldsymbol{\mathcal{P}}_i^{\mathrm{I}} - \boldsymbol{\mathcal{P}}_j^{\mathrm{I}}\right)\end{aligned} \tag{11}$$

where $\boldsymbol{P}_i^{\mathrm{I}}$ and $\boldsymbol{P}_j^{\mathrm{I}}$ are respectively the current position vectors for SSUH i and SSUH j ; $\boldsymbol{P}_{i0}^{\mathrm{I}}$ and $\boldsymbol{P}_{j0}^{\mathrm{I}}$ are respectively the desired position vector pointing to SSUH i and SSUH j in the formation. It is assumed that all the mutual influence force gains, k_{ij}, $i,j=1,\ldots,n$, are identical, and let $k_{ij} = k_{mut} > 0$. In the swarm of N_{SSUH} SSUHs, the total mutual influencing force for SSUH i is given by

$$\boldsymbol{F}_i^{mut} = \sum_{j=1}^{N_{SSUH}} \boldsymbol{F}_{ij} \tag{12}$$

This force preserves the formation by affecting the helicopters to remain in their desired distances among themselves.

C. *Collision-Avoidance Force*

For safety issue, it is necessary to set up a safety space which is defined for each helicopter. For the sake of simplicity, it is defined as a sphere with positive radius $r_{sav} = r_{sav}^{\min}$, where $r_{sav}^{\min}$ is the minimum distance for save avoidance. The collision-avoidance force is expressed by

$$\boldsymbol{F}_{ij}^{ca} = \begin{cases} k_{ca} \times \left(\dfrac{r_{sav}}{\|\boldsymbol{d}_{ji}\|} - 1\right) \times \dfrac{\boldsymbol{d}_{ji}}{\|\boldsymbol{d}_{ji}\|} & \text{for } \|\boldsymbol{d}_{ji}\| < r_{sav} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

where $\mathscr{d}_{ji}$ is the vector between SSUH i and j.

Obviously, the collision-avoidance force will be generated

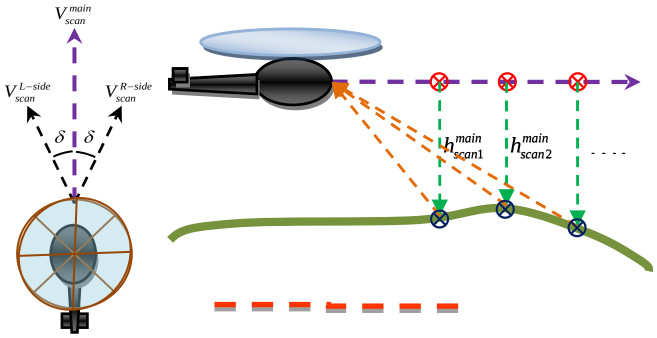


Fig. 4. Obstacle detection of the sensor mounted in front of each helicopter.

if something enters the sphere, pointing away from the invading SSUH. To ensure collision avoidance, it is required to let the collision-avoidance force go toward infinity in the center of this sphere. Oppositely, if no something others into the sphere, the collision-avoidance force will become zero. Therefore the collision-avoidance for SSUH i whose safety sphere away from others is defined by $\|\bullet\|_2$ represents the vector 2-norm throughout the paper. If $\|\boldsymbol{d}_{ji}\| \geq r_{sav}$ the collision-avoidance force becomes zero. This allows a smooth insertion of the collision avoidance force component and guarantees a continuous potential field. Moreover, k_{ca} is the collision-avoidance gain to be tuned. To enhance the collision-avoidance performance, the safety distance r_{sav} should be chosen properly, depending on the helicopter's velocity, $\|\dot{\boldsymbol{P}}^{\mathrm{I}}\|$, and it is expressed by

$$r_{sav} = r_{sav}^{\min} + k_{sav} \times \|\dot{\boldsymbol{P}}^{\mathrm{I}}\| \tag{14}$$

where k_{sav} is the safety gain and $r_{sav}^{\min}$ is the minimum distance for a save avoidance. Thus, the total amount of the collision avoidance term is given by:

$$\boldsymbol{F}_i^{ca} = \sum_{j=1,\, j\neq i}^{N} \boldsymbol{F}_{ij}^{ca} \tag{15}$$

D. *Obstacle-Avoidance Force*

During the contour flight, it is often easy to encounter with some obstacles which may be unpredictable terrain, buildings, trees, or other aircraft. Under these obstacles, the helicopters will consider them harmful, especially in tactical flight situation. Therefore, it is necessary to shift out these threat points by building up enough repulsion force, thus making UAV formation flight safe.

The obstacle can be detected in terms of distances and angles by the sensor mounted in front of each helicopter. Note that the sensor has its physical limitations on detecting the obstacle.

$$\hat{\mathbf{F}}_{ik}^{oa} = \begin{cases} \dfrac{k_{oa}}{\left(h_{scan}^{Line}\right)^2} \times \left(\dfrac{1}{\|\mathscr{d}_{ki}\|} - \dfrac{1}{r_{sav}}\right) \times \dfrac{\mathscr{d}_{ki}}{\|\mathscr{d}_{ki}\|} & \text{for } \|\mathscr{d}_{ki}\| < r_{sav} \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

where r_{sav} is given in (14); k_{oa} is the gain should be found, h_{scan}^{Line} stands for terrain high of detected obstacle at pre-navigation point that front of the SSUH and $\mathbf{d}_{ki}$ represents the distance between SSUH *i* and the kth obstacle. As shown in Fig. 4, the sensor continuously monitor the foreground from its right starting line, V_{scan}^{R-side}, to the left one, V_{scan}^{L-side}, where δ is the angle from the flight direction line, V_{scan}^{mian}. Afterward, the height of the detected obstacles can be found. By on-line scanning obstacles in front of each helicopter, we propose the subsequent obstacle-avoidance force and the total obstacle-avoidance force for each helicopter, assuming that there are detected M , W and W obstacles at proposed scan line V_{scan}^{mian}, V_{scan}^{R-side} and V_{scan}^{L-side}, respectively. Finally, the total forces of detected obstacles are expressed by

$$\mathbf{F}_i^{oa} = \sum_{k=1}^{M} \mathbf{F}_{main}^{oa}(i,k) + \sum_{z=1}^{W} \mathbf{F}_{L-side}^{oa}(i,z) + \sum_{z=1}^{W} \mathbf{F}_{R-side}^{oa}(i,z) \qquad (17)$$

E. Total MPF Forces

By summing the aforementioned force components which meet the constraints for helicopter flight control, one obtains the magnitude and direction of the continuously total forces for SSUH i at its present position.

$$\bar{\boldsymbol{F}}_i = \boldsymbol{F}_i^{VL} + \boldsymbol{F}_i^{VVL} + \boldsymbol{F}_i^{mut} + \boldsymbol{F}_i^{ca} + \boldsymbol{F}_i^{oa} \qquad (18)$$

Due to the physical limitations of each SSUH, it is necessary to define the maximum amplitude for the total force vector for each SSUH while keeping its direction:

$$\boldsymbol{F}_i = \min\left\{ \left\| \bar{\boldsymbol{F}}_i \right\|, F_{\max} \right\} \bar{\boldsymbol{F}}_i \Big/ \left\| \bar{\boldsymbol{F}}_i \right\| \qquad (19)$$

where $F_{\max}$ will be the upper limit regarding the total MPF force's strength and therefore a limitation for the SSUH's speed. $F_{\max}$ should be chosen dynamically to use the maximum SSUH speed. This can be realized by taking the SSUH's NED (North-East-Down) velocity $\left\| \dot{\boldsymbol{P}}^{\mathrm{I}} \right\|$ into account:

$$F_{\max} = F_{\min} + k_v \left\| \dot{\boldsymbol{P}}^{\mathrm{I}} \right\| \qquad (20)$$

where $F_{\min}$ is the minimum value and k_v is the gain. When one SSUH is accelerating, the distance to the vehicle's reference position will also increase, thereby speeding up the SSUH until its maximal velocity is reached. Once the total force vector $\bar{\boldsymbol{F}}_i$ has been obtained from (18), the reference trajectory for each SSUH is calculated by

$$\boldsymbol{P}_i^{\mathrm{I}}(t+\Delta t) = \boldsymbol{P}_i^{\mathrm{I}}(t) + \bar{\boldsymbol{F}}_i \, \Delta t \qquad (21)$$

where Δt is the step size. On the other hand, since $\bar{\boldsymbol{F}}_i \Big/ \left\| \bar{\boldsymbol{F}}_i \right\|$ represents for the direction of SSUH *i*, the heading angle of the i^{th} SSUH at time t is given by $\psi_{d_i}(t) = (\bar{\boldsymbol{F}}_i \Big/ \left\| \bar{\boldsymbol{F}}_i \right\|) \times (0 \quad 0 \quad 1)^T$.

IV. Exponential Stability and Consensus Tracking

This section will show exponential consensus tracking of the generated formation trajectories of the proposed MPF trajectory generator if $\bar{\boldsymbol{F}}_i = \boldsymbol{F}_i^{VL} + \boldsymbol{F}_i^{VVL} + \boldsymbol{F}_i^{mut} + \boldsymbol{F}_i^{ca} + \boldsymbol{F}_i^{oa}$ is given by considering the moving speed of the leader. In doing so, we first prove that the generated formation trajectories for all the SSUHs in formation is exponentially stable if $\bar{\boldsymbol{F}}_i = \boldsymbol{F}_i^{VL} + \boldsymbol{F}_i^{VVL} + \boldsymbol{F}_i^{mut}$ is applied, and then discuss the effects of the collision and obstacle–avoidance forces on the formation trajectories. This is because the summation of the first two forces affects the formation stability, while the collision and obstacle–avoidance forces circumvent any possible collisions. To show the formation stability, we rewrite (21) as $\dot{\boldsymbol{P}}_i^{\mathrm{I}} = \lim_{\Delta t \to 0} (\boldsymbol{P}_i^{\mathrm{I}}(t+\Delta t) - \boldsymbol{P}_i^{\mathrm{I}}(t)) / \Delta t = \bar{\boldsymbol{F}}_i$ or $\dot{\boldsymbol{P}}^{\mathrm{I}}(t) = \bar{\boldsymbol{F}}$ where $\dot{\boldsymbol{P}}^{\mathrm{I}} = [(\dot{\boldsymbol{P}}_1^{\mathrm{I}})^T, \cdots, (\dot{\boldsymbol{P}}_n^{\mathrm{I}})^T]^T$ and $\bar{\boldsymbol{F}} = [\bar{\boldsymbol{F}}_1^T, \cdots, \bar{\boldsymbol{F}}_n^T]^T$. Below is the main result for considering the issue by focusing on the impact of the only first three forces on formation stability.

Theorem 1: Assume that the FBFN-based trajectory tracking controllers for all SSUHs in formation are semi-globally uniformly ultimately bounded (SUUB). Then the generated formation trajectories of the proposed MPF trajectory generator for all the SSUHs in formation is exponentially stable and the asymptotical consensus tracking is satisfied if the errors of the generated trajectories meet $\dot{\boldsymbol{P}}_i^{\mathrm{I}} = \bar{\boldsymbol{F}}_i$ where $\bar{\boldsymbol{F}}_i = \boldsymbol{F}_i^{VL} + \boldsymbol{F}_i^{VVL} + \boldsymbol{F}_i^{mut}$.

Proof: To show the formation stability of the generated trajectories by the formula $\dot{\boldsymbol{P}}_i^{\mathrm{I}}(t) = \bar{\boldsymbol{F}}_i$, $i = 1, \ldots, n$, or $\dot{\boldsymbol{P}}^{\mathrm{I}}(t) = \bar{\boldsymbol{F}}$ one defines the position tracking error vector for the system as

$$\boldsymbol{e}_i = \sum_{j=1}^{n+1} a_{ij} (\boldsymbol{\mathcal{P}}_i - \boldsymbol{\mathcal{P}}_j) \qquad (22)$$

which leads to

$$\boldsymbol{e} = (\boldsymbol{L} + \boldsymbol{B}) \otimes \boldsymbol{I}_3 (\boldsymbol{P}^{\mathrm{I}} - \boldsymbol{P}_0^{\mathrm{I}}) \qquad (23)$$

where $\boldsymbol{P}_0^{\mathrm{I}} \equiv ((\boldsymbol{P}_{10}^{\mathrm{I}})^T, \ldots, (\boldsymbol{P}_{n0}^{\mathrm{I}})^T)^T \in R^{3n}$ and $\boldsymbol{P}^{\mathrm{I}} \equiv ((\boldsymbol{P}_1^{\mathrm{I}})^T, \ldots, (\boldsymbol{P}_n^{\mathrm{I}})^T)^T \in R^{3n}$. Taking the time derivative of the position tracking error vector $\boldsymbol{e}$ gives

$$\dot{\boldsymbol{e}} = (\boldsymbol{L} + \boldsymbol{B}) \otimes \boldsymbol{I}_3 (\dot{\boldsymbol{P}}^{\mathrm{I}} - \dot{\boldsymbol{P}}_0^{\mathrm{I}}) = (\boldsymbol{L} + \boldsymbol{B}) \otimes \boldsymbol{I}_3 (\bar{\boldsymbol{F}} - \dot{\boldsymbol{P}}_0^{\mathrm{I}}).$$

To show the stability of the generated formation trajectories, we rewrite the summation of the virtual leading force and mutual influencing forces of the i^{th} SSUH in formation as

$$\bar{\boldsymbol{F}}_i = \dot{d}_i + \sum_{j=1}^{n} \bar{l}_{ij} b_j \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}} - \sum_{j=1}^{n+1} a_{ij} k_{ij} (\boldsymbol{\mathcal{P}}_i^{\mathrm{I}} - \boldsymbol{\mathcal{P}}_j^{\mathrm{I}}), \; i = 1, \ldots, n \qquad (24)$$

Note that the virtual leader is regarded as the $(n+1)^{th}$

SSUH, and $k_{i(n+1)} = k_{VL}$, $i = 1, \dots, n$. Thus, the total force control vector $\bar{\boldsymbol{F}}_i$ can be rewritten in a vector-matrix form.

$$\bar{\boldsymbol{F}} = \dot{\mathbb{d}} + ((\boldsymbol{L}+\boldsymbol{B})^{-1} \otimes \boldsymbol{I}_3)(\boldsymbol{b} \otimes \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}}) - k_{mut}(\bar{\boldsymbol{L}} \otimes \boldsymbol{I}_3)\bar{\boldsymbol{P}}^{\mathrm{I}} \quad (25)$$

where

$\bar{\boldsymbol{F}} = (\boldsymbol{F}_1^T, ..., \boldsymbol{F}_n^T, \boldsymbol{0}_3)^T$, $\bar{\boldsymbol{P}}^I \equiv (\boldsymbol{\mathcal{P}}_1^T, ..., \boldsymbol{\mathcal{P}}_n^T, \boldsymbol{\mathcal{P}}_{n+1}^T)^T \in R^{3(n+1)}$; $\otimes$ denotes the Kronecter product ; the graph Laplacian $\bar{\boldsymbol{L}} \in R^{(n+1)\times(n+1)}$ of the formation network composed of the n SSUHs and leader is

$$\bar{\mathbf{L}} = \begin{bmatrix} \frac{k_{VL}}{k_{mut}} a_{1(n+1)} + \sum_{j=1}^{n} a_{1j} & \cdots & -a_{1n} & -a_{1(n+1)} \\ -a_{2n} & \frac{k_{VL}}{k_{mut}} a_{2(n+1)} + \sum_{j=1}^{n} a_{2j} & \cdots & -a_{2(n+1)} \\ \vdots & \ddots & \vdots & \vdots \\ -a_{n1} & \cdots & \frac{k_{VL}}{k_{mut}} a_{n(n+1)} + \sum_{j=1}^{n} a_{nj} & -a_{n(n+1)} \\ 0 & \cdots & 0 & 0 \end{bmatrix} \quad (26)$$

where $a_{ii} = 0$ and $a_{i(n+1)} \sum_{j=1}^{n+1} a_{ij} - a_{i1} - a_{i2} - \cdots - a_{in}$ for $i = 1, \dots, n$. Since $\boldsymbol{d}_{n+1}(t) = 0$ for any time, then, by following the proof process, it is easy to rewrite the total force control vector as follows;

$$\begin{aligned} \bar{\boldsymbol{F}} &= \dot{\mathbb{d}} + ((\boldsymbol{L}+\boldsymbol{B})^{-1} \otimes \boldsymbol{I}_3)(\boldsymbol{b} \otimes \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}}) - k_{mut}(\boldsymbol{T} \otimes \boldsymbol{I}_3)(\boldsymbol{P}^{\mathrm{I}} - \boldsymbol{P}_0^{\mathrm{I}}) \\ &= \dot{\mathbb{d}} + ((\boldsymbol{L}+\boldsymbol{B})^{-1} \otimes \boldsymbol{I}_3)(\boldsymbol{b} \otimes \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}}) - k_{mut}(\boldsymbol{T} \otimes \boldsymbol{I}_3)(\boldsymbol{L}+\boldsymbol{B})^{-1} \otimes \boldsymbol{I}_3\boldsymbol{e} \quad (27) \\ &= \dot{\mathbb{d}} + ((\boldsymbol{L}+\boldsymbol{B})^{-1} \otimes \boldsymbol{I}_3)(\boldsymbol{b} \otimes \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}}) - k_{mut}\boldsymbol{T}(\boldsymbol{L}+\boldsymbol{B})^{-1} \otimes \boldsymbol{I}_3\boldsymbol{e} \end{aligned}$$

where $\bar{\boldsymbol{F}} = (F_{\boldsymbol{1}}^{\boldsymbol{T}}, F_{\boldsymbol{2}}^{\boldsymbol{T}}, ..., F_{\boldsymbol{n}}^{\boldsymbol{T}})^T \in R^{3n}$ the matrix $\boldsymbol{T}$ is reduced from the graph Laplacian $\bar{\boldsymbol{L}}$ and given by

$$\mathbf{T} = \begin{bmatrix} \frac{k_{VL}}{k_{mut}} a_{1(n+1)} + \sum_{j=1}^{n+1} a_{1j} & -a_{12} & \cdots & -a_{1n} \\ -a_{2n} & \frac{k_{VL}}{k_{mut}} a_{2(n+1)} + \sum_{j=1}^{n+1} a_{2j} & \cdots & -a_{2n} \\ \cdots & \cdots & \ddots & \vdots \\ -a_{n1} & -a_{n2} & \cdots & \frac{k_{VL}}{k_{mut}} a_{n(n+1)} + \sum_{j=1}^{n+1} a_{nj} \end{bmatrix} \quad (28)$$

Note that the matrix $\boldsymbol{T} = \boldsymbol{L} + k_{VL}\boldsymbol{B} / k_{mut}$ is invertible and has all the nonzero eigenvalues with positive real parts, as mentioned in Section 2.4. With the total force control vector in (27), the error dynamics of the generated trajectory error becomes

$$\begin{aligned} \dot{\boldsymbol{e}} &= (\boldsymbol{L}+\boldsymbol{B}) \otimes \boldsymbol{I}_3(\bar{\boldsymbol{F}} \ - \dot{\boldsymbol{P}}_0^{\mathrm{I}}) = (\boldsymbol{L}+\boldsymbol{B}) \otimes \boldsymbol{I}_3(\bar{\boldsymbol{F}} \ - \dot{\boldsymbol{P}}_0^{\mathrm{I}}) \\ &= (\boldsymbol{L}+\boldsymbol{B}) \otimes \boldsymbol{I}_3(\dot{\mathbb{d}} + ((\boldsymbol{L}+\boldsymbol{B})^{-1} \otimes \boldsymbol{I}_3)(\boldsymbol{b} \otimes \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}}) - k_{mut}\boldsymbol{T}(\boldsymbol{L}+\boldsymbol{B})^{-1} \otimes \boldsymbol{I}_3\boldsymbol{e} - \dot{\boldsymbol{P}}_0^{\mathrm{I}}) \end{aligned} \quad (29)$$

Since $\dot{\mathbb{d}} - \dot{\boldsymbol{P}}_0^{\mathrm{I}} = -\boldsymbol{1} \otimes \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}}$ where $\boldsymbol{1} = [1, \cdots, 1]^T \in R^n$, it follows that $(\boldsymbol{L}+\boldsymbol{B})\boldsymbol{1} = \boldsymbol{L1} + \boldsymbol{B1} = 0 + \boldsymbol{b} = \boldsymbol{b}$, and (29) becomes

$$\begin{aligned} \dot{\boldsymbol{e}} &= \cancel{-\boldsymbol{b} \otimes \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}}} \ \cancel{+\boldsymbol{b} \otimes \dot{\boldsymbol{P}}_{VL}^{\mathrm{I}}} - k_{mut}((\boldsymbol{L}+\boldsymbol{B}) \otimes \boldsymbol{I}_3)(\boldsymbol{T}(\boldsymbol{L}+\boldsymbol{B})^{-1} \otimes \boldsymbol{I}_3\boldsymbol{e} \\ &= -k_{mut}(\boldsymbol{L}+\boldsymbol{B})\boldsymbol{T}(\boldsymbol{L}+\boldsymbol{B})^{-1} \otimes \boldsymbol{I}_3\boldsymbol{e} \end{aligned} \quad (30)$$

Next, it is easy to show that the product matrix $(\boldsymbol{L}+\boldsymbol{B})\boldsymbol{T}(\boldsymbol{L}+\boldsymbol{B})^{-1}$ is also invertible and has all the nonzero eigenvalues with positive real parts. Therefore, exist a symmetric and positive-definite matrix $\boldsymbol{Z}$, and a symmetric and diagonal positive-definite matrix $\boldsymbol{R}$ such that $\boldsymbol{R}((\boldsymbol{L}+\boldsymbol{B})\boldsymbol{T}(\boldsymbol{L}+\boldsymbol{B})^{-1}) \quad +((\boldsymbol{L}+\boldsymbol{B})\boldsymbol{T}(\boldsymbol{L}+\boldsymbol{B})^{-1})^T \boldsymbol{R} = \boldsymbol{Z}$. Thus, we choose the following positive-definite and radially unbounded Lyapunov function V_1 as

$$V_1 = (\boldsymbol{e})^T (\boldsymbol{R} \otimes \boldsymbol{I}_3)(\boldsymbol{e}) / 2 \quad (31)$$

Taking the time derivative of (31) along the trajectories of (27) , one obtains

$$\begin{aligned} \dot{V}_1 &= -k_{mut}(\boldsymbol{e})^T(\boldsymbol{R}((\boldsymbol{L}+\boldsymbol{B})\boldsymbol{T}(\boldsymbol{L}+\boldsymbol{B})^{-1}) + ((\boldsymbol{L}+\boldsymbol{B})\boldsymbol{T}(\boldsymbol{L}+\boldsymbol{B})^{-1})^T\boldsymbol{R} \otimes \boldsymbol{I}_3)(\boldsymbol{e}) \\ &= -k_{mut}(\boldsymbol{e})^T(\boldsymbol{Z} \otimes \boldsymbol{I}_3)(\boldsymbol{e}) < 0 \end{aligned} \quad (32)$$

which is negative-definite, showing that the error of the generated trajectory approaches zero as time goes to infinity, namely that $\lim_{t\to\infty} \boldsymbol{e} = 0$. From (23), one obtains $\lim_{t\to\infty} \boldsymbol{P}^{\mathrm{I}} = \boldsymbol{P}^{\mathrm{I}} - \boldsymbol{P}_0^{\mathrm{I}} = 0$, i.e., $\lim_{t\to\infty}(\boldsymbol{P}_i^{\mathrm{I}} - \boldsymbol{P}_{i0}^{\mathrm{I}}) = 0$ since the matrix $\boldsymbol{L}+\boldsymbol{B}$ is invertible. This means that all the n follower SSUHs converge to their desired position specified by the virtual leader, thus forming a stable formation pattern even if the leader is moving at the speed vector of $\dot{\boldsymbol{P}}_{VL}^{\mathrm{I}}$. Once the stable formation pattern has been generated by the force vector $\bar{\boldsymbol{F}}_i = \boldsymbol{F}_i^{VL} + \boldsymbol{F}_i^{VVL} + \boldsymbol{F}_i^{mut}$, it is easy to show that $\lim_{t\to\infty} \boldsymbol{\mathcal{P}}_i^{\mathrm{I}} - \boldsymbol{\mathcal{P}}_j^{\mathrm{I}} = \boldsymbol{P}_{VL}^{\mathrm{I}} - \boldsymbol{P}_{VL}^{\mathrm{I}} = 0$, $i = 1, \dots, n$ and $j = 1, \dots, n$, thereby achieving the globally exponential consensus tracking. This completes the proof.

Remark 1: Since each SSUH may not get its desired position from the leader due to the communication topology, the generated force vector $\bar{\boldsymbol{F}}_i = \boldsymbol{F}_i^{VL} + \boldsymbol{F}_i^{VVL} + \boldsymbol{F}_i^{mut}$ meets the requirement and it is distributed and decentralized. Furthermore, the overall force vector $\bar{\boldsymbol{F}}_i = \boldsymbol{F}_i^{VL} + \boldsymbol{F}_i^{VVL} + \boldsymbol{F}_i^{mut} + \boldsymbol{F}_i^{ca} + \boldsymbol{F}_i^{oa}$ is also distributed and decentralized owing to the fact that the collision-avoidance and obstacle-avoidance forces are made by the i^{th} SSUH that detects any local SSUHs and/or obstacles nearby itself.

Remark 2: The exponential stability of the generated formation pattern is still guaranteed if the overall $\bar{\boldsymbol{F}}_i = \boldsymbol{F}_i^{VL} + \boldsymbol{F}_i^{VVL} + \boldsymbol{F}_i^{mut} + \boldsymbol{F}_i^{ca} + \boldsymbol{F}_i^{oa}$ is applied and if the destination is **far** from any obstacles. Obviously, in some cases, if all the SSUHs fly together in formation and there are no obstacles surrounding them, then the last two forces die out, and the first three forces will bring all the SSUHs to reach their desired position and exponential consensus condition as predicted by Theorem 1. Note that, in some cases, the total magnitude of the collision and obstacle force will be equal to the attraction force summed by $\boldsymbol{F}_i^{VL} + \boldsymbol{F}_i^{VVL} + \boldsymbol{F}_i^{mut}$, but the direction is opposite, thereby causing a dead-locking problem such that the exponential consensus tracking is not achieved.

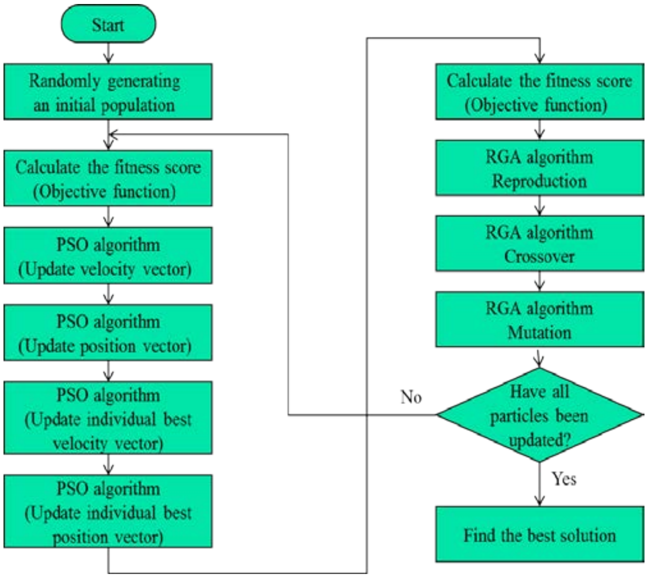

Start
Randomly generating an initial population
Calculate the fitness score (Objective function)
PSO algorithm (Update velocity vector)
PSO algorithm (Update position vector)
PSO algorithm (Update individual best velocity vector)
PSO algorithm (Update individual best position vector)
Calculate the fitness score (Objective function)
RGA algorithm Reproduction
RGA algorithm Crossover
RGA algorithm Mutation
Have all particles been updated?
No
Yes
Find the best solution

Fig. 5. PSO-RGA computing procedure.

This problem can be circumvented by using a modified collision and obstacle force vector approach, and utilizing a Lyapunov function to show the exponential stability of the overall formation system. However, the method requires a full connected communication topology, which is not suitable for any directed communication topology. Another alternative is to use periodic switching topology method which does not affect the exponential stability, but may avoid the dead-locking problem.

V. Parameter Tuning of the PF Formation Controller Using PSO-RGA

This section is devoted to applying the PSO-RGA algorithm to optimally search for the three best parameters, k_{VL}, k_{ij}=k_{mut} and k_v, of the MPF-based trajectory generator for formation control of the multi-helicopter system. Particle swarm optimization (PSO) is known as a kind of evolutionary computation algorithm based on swarm intelligence method. The basic concept of PSO comes from simulation of social behavior and evolutionary computation theory, and its basic working idea is that m particles in the D-dimensional searching space are attracted towards the optimal solution found by given particles' neighbors and by the particles' cognition and experience when those particles work through their searching space.

On the other hand, real-coded genetic algorithms (RGA) have been regarded as adaptive, heuristic and biological search methods with four basic operators: selection, reproduction, crossover and mutation. Real-coded genetic algorithms (RGA) are different from binary genetic algorithms (BGA) due to no coding and decoding computational requirements for all chromosomes. The key idea behind RGA is to construct the fitness function to evaluate all chromosomes in a population, and to choose the best chromosomes with the best fitness in order to reproduce new chromosomes via the selection operator. The used PSO-RGA algorithm takes the advantages of aforementioned PSO and RGA algorithms.

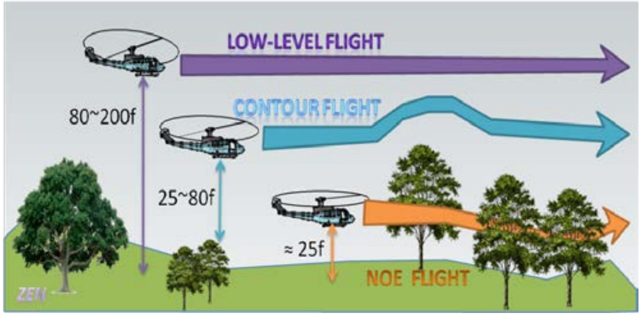


Fig. 6. Illustration of low-level, contour and NOE flights for the multi-SSUH system.

Since simultaneous tuning of the three parameters in the formation controller is difficult via a try-and-error rule, the PSO-RGA algorithm proposed in [25] will be used to reduce design time and effort, and even obtain improved formation control performance. The used PSO-RGA algorithm takes the advantages of aforementioned PSO and RGA algorithms. Fig. 5 shows the detailed flow chart of the used PSO-RGA algorithm which is described as follows.

Step 1: Update the PSO velocity strategy. Compute the fitness value of each individual in the population. Compare the position value of $X_i(t)$ for each particle with the value resulting from its current evaluation. $X_i(t)$ is replaced if the latter value is better; otherwise, retain it. Finally, take $X_i(t)$ as the particle's current location.

Step 2: Update the PSO position strategy. Update the optimal local position $P_i(t)$ of each particle and the best global position. If the new position is outside the feasible space, set the position of the particle at the margin of the feasible space nearest the new position by the following rule.

$$\begin{aligned} &\text{if } rand < mr, x_{iD}(t+1) = x_{iD}(t) + \left[v_{iD}(t+1)/2N(0,1)\right] \\ &\text{else }, x_{iD}(t+1) = x_{iD}(t) + \left[v_{iD}(t+1)/\sigma\right] \end{aligned} \tag{33}$$

where $rand$ is a uniformly distributed random generator giving its value lying within the close interval [0,1]; mr represents the selected mutation rate; N(0, 1) is a standard Gaussian random variable; $1 \le \sigma < 2, \sigma \in R$ is the deceleration factor of the particle *i*.

Step 3: Execute RGA reproduction policy using $P_r = f_i / \sum_{i=1}^{l} f_i$ where f_i is the fitness value of the i^{th} individual chromosome, and l is the size of group. For each particle j, use the aforementioned roulette wheel selection technique to choose x_{jD} from the set of all X_j.

Step 4: Perform the crossover operation with the crossover rate Pc using

$$\bar{x}_1 = x_1 - \lambda(x_1 - x_2), \bar{x}_2 = x_2 + \lambda(x_1 - x_2) \tag{34}$$

where $\bar{x}_1$ and $\bar{x}_2$ are the child chromosomes; x_i and x_2 are respective parent chromosomes; λ is a uniformly distributed random number such that $\lambda \in [0,1]$.

Step 5: Do the RGA mutation strategy with the mutation rate Pm using

$$x_{jD}(t+1) = x_{jD}(t) + N\left(0, \beta f_i / f_\Sigma\right) \qquad (35)$$

where $x_{jD}(t)$ represents the D^{th} entry of the j^{th} individual chromosome; $N\left(0,\beta f_i / f_\Sigma\right)$ denotes a Gaussian random variable with zero mean and standard deviation of $\beta f_i / f_\Sigma$; f_Σ is the summation of all computed fitness values; and β is the scaling parameter for f_i / f_Σ.

Step 6: Repeat Step 1 until the following cost $f(E)$ in (36) is minimized, and then evaluate it to see if $f(E)$ is no longer evaluated.

$$\begin{aligned}\min f\left(k_{VL},k_{ij},k_v\right) &= \alpha_v \times \left(\sum_{i=1}^{N}\sum_{\Delta t} \textit{the } i^{th} \textit{ vehicle speed}\right) \\ &+ \alpha_a \times \left(\sum_{i=1}^{N}\sum_{\Delta t} \textit{the } i^{th} \textit{ vehicle tuning angle}\right)\end{aligned} \qquad (36)$$

VI. SIMULATIONS AND DISCUSSIONS

This section will conduct four simulations to examine the effectiveness, performance and merit of the proposed MPF-based formation control system for a group of four cooperative helicopters with a virtual leader, in order to achieve contour flight in formation, as Fig. 6 shows. Fig.6 depicts the contour flight that keeps 25 to 80 foots high above the flight terrain. The environment is a complicated, rough terrain with a space area of 1km x 1km as a three-dimensional flight terrain. To verify the feasibility of the proposed control method, we adopt the well-done nonlinear helicopter model in [27] and the FBFN-based controller and its key control parameters in [26]. For the simulations, the parameter settings of the PSO-RGA algorithm are given by mr =0.35, β =0.001, σ =1, α_a =1, α_v =1, Pc=0.5, and Pm=0.5. During the simulations, the maximum speed and turning angle of each SSUH is less than 10 m/sec, and less than 30 degrees/sec, respectively. In these simulations, the leader's trajectory is generated by the presented RGA algorithm and elastic-band techniques in [30], and the resultant trajectory is smoothened via the spline curve fitting scheme and then used as the final flight trajectory. Fig. 2(a) illustrates the used communication topology which has the least information exchange, and, thus, we have the graph Laplacian $\bar{\boldsymbol{L}}$ and reduced graph Laplacian $\boldsymbol{T} = \boldsymbol{L} + k_{VL}\boldsymbol{B} / k_{mut}$ where $\boldsymbol{B}$ =diag{1,0,0,0} and $k_{VL} = 0.4974$ and $k_{mut} = k_{ij} = 0.1254$ as shown in Table 1.

TABLE I. THE BEST SEARCHED PARAMETERS AND THE PARAMETERS IN [9].

MPF Parameters	Proposed method	Paul *et al* [9]	Descriptions
k_{VL}	0.497355572952685 †	1	Virtual leader gain
k_{ij}	0.125403575307529 †	0.1	Inter vehicle gain
k_v	0.700000000000000 †	unavailable	Vehicle speed gain
k_{ca}	2 * k_{VL} * $r_{sav}^{\min}$	240	Collision avoidance gain
k_{oa}	40	unavailable	Obstacle avoidance gain
$r_{sav}^{\min}$	10m	30m	Safety radius
k_{sav}	1	unavailable	Safety radius gain

† The parameters are searched by the PSO-RGA method.

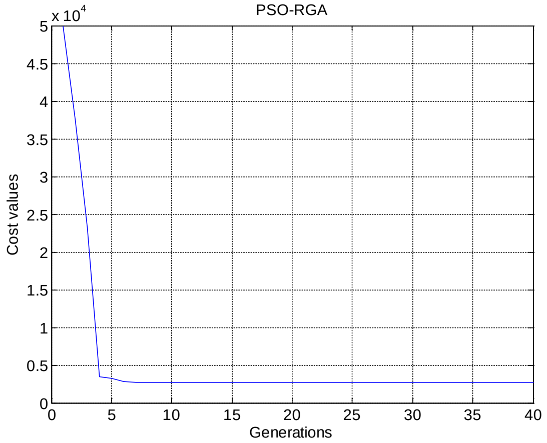


Fig. 7. The cost during the simulation over the 40 generations.

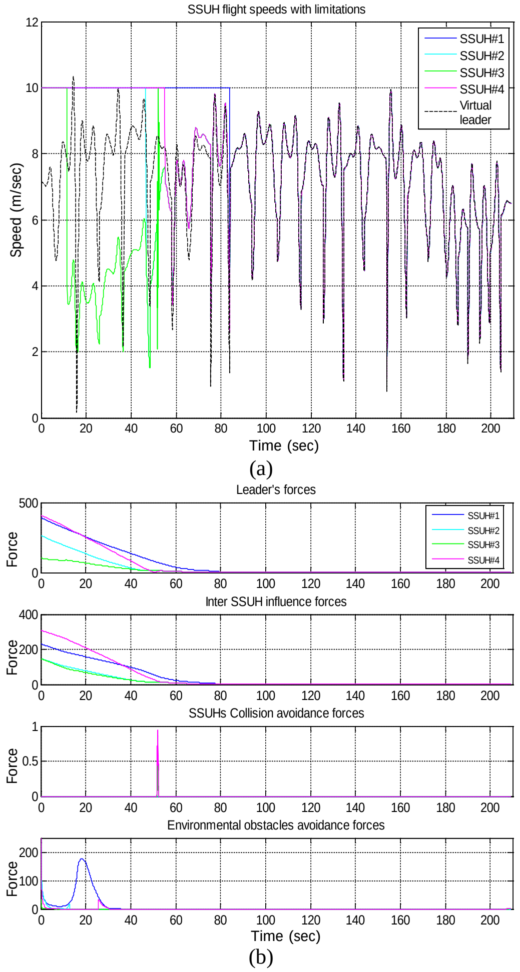
$$\bar{\boldsymbol{L}} = \begin{bmatrix} 1 & 0 & 0 & 0 & -1 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{T} = \begin{bmatrix} 3.9665 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

Both symmetric and positive-definite matrices $\boldsymbol{R}$ and $\boldsymbol{S}$, satisfying $\boldsymbol{RT} + \boldsymbol{T}^T\boldsymbol{R} = \boldsymbol{S}$, are found as follows;

$$\mathbf{R} = \begin{bmatrix} 3.9665 & 0 & 0 & 0 \\ 0 & 0.7987 & 0 & 0 \\ 0 & 0 & 0.4440 & 0 \\ 0 & 0 & 0 & 0.3075 \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} 31.4662 & -0.7987 & 0 & 0 \\ -0.7987 & 1.5973 & -0.4440 & 0 \\ 0 & -0.4440 & 0.8881 & -0.3075 \\ 0 & 0 & -0.3075 & 0.6150 \end{bmatrix}$$

which ensure the asymptotical stability of the generated formation pattern, thus achieving consensus tracking.

The first simulation is to search for the best three parameters of the MPF-based trajectory generator using the PSO-RGA algorithm with 40 generations, each of which has 20 particles. The first simulation uses the horizontal line formation pattern where $\boldsymbol{d}_1$ =(0,15,0), $\boldsymbol{d}_2$ =(0,35,0), $\boldsymbol{d}_3$ =(0,-15,0) and $\boldsymbol{d}_4$ =(0,-35,0) (unit:m). After the off-line simulation, the best three parameters are found as $k_{VL} = 0.4973$, k_{ij} =0.1254 and k_v =0.7. Fig. 7 depicts the cost with the 40 generations, reaching its minimum after the fourth generation. Table 1 lists the best searched values of the key parameters, in comparison with the same parameters given in [9]. Fig. 8 illustrates the simulated trajectories and speeds of the four follower SSUHs over the terrain; Fig. 8(a) shows that



(a)

(b)

Fig. 8. (a) Simulated speeds of all the four SSUHs; (b) all the four forces of the follower helicopters .

the speeds of the four following helicopter converge to the given speed of the leader, and Fig. 8(b) demonstrates the time evolutions of all four forces, indicating that the collision-avoidance and obstacle-avoidance forces hold in order to circumvent any possible collisions. Fig. 9 displays the simulated trajectories, speeds and all the four forces of the four following helicopters using the parameters in [9], in which all the speeds are not conformed to the leader's one. Via the performance comparison of the results in Figs. 8 and 9, the proposed MPF method is shown more effective and superior in terms of formation control over the terrain.

The second simulation is to examine whether the proposed distributed consensus MPF formation control system with the developed FBFN trajectory tracking controller together is capable of tracking the generated formation pattern and trajectories generated by the MPF method over the complicated and rough terrain as shown in Fig. 10. Given initial position of the four following helicopters respectively as (101.7 762.7 31.16), (50.85 491.5 51.39), (186.4 101.7 34.64), and (813.6 84.75 45.11) (unit:m), Fig. 10 depicts the simulated trajectories of the proposed formation control system. Figs. 10(b)-(e) depict the tracking position errors of the four SSUHs by using the Euclidean distance measure,

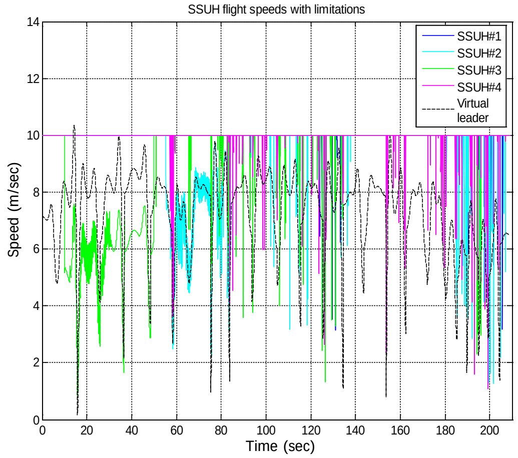
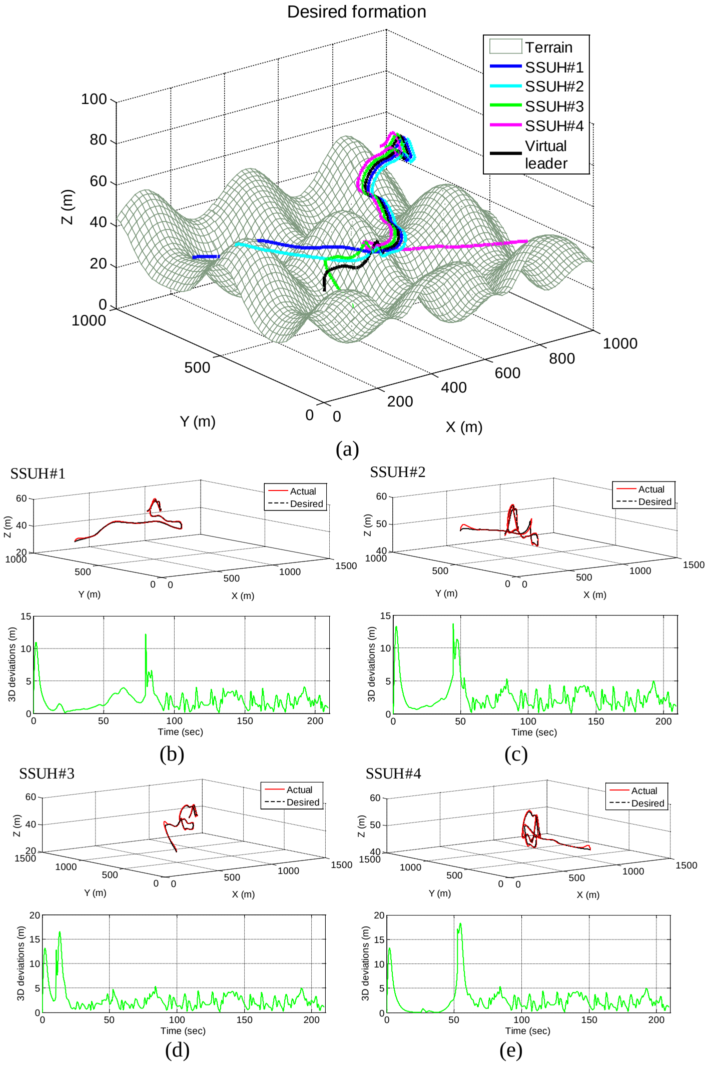


Fig. 9. Simulated speeds of all the four SSUHs using the PF parameters in [9].



(a)

(b)

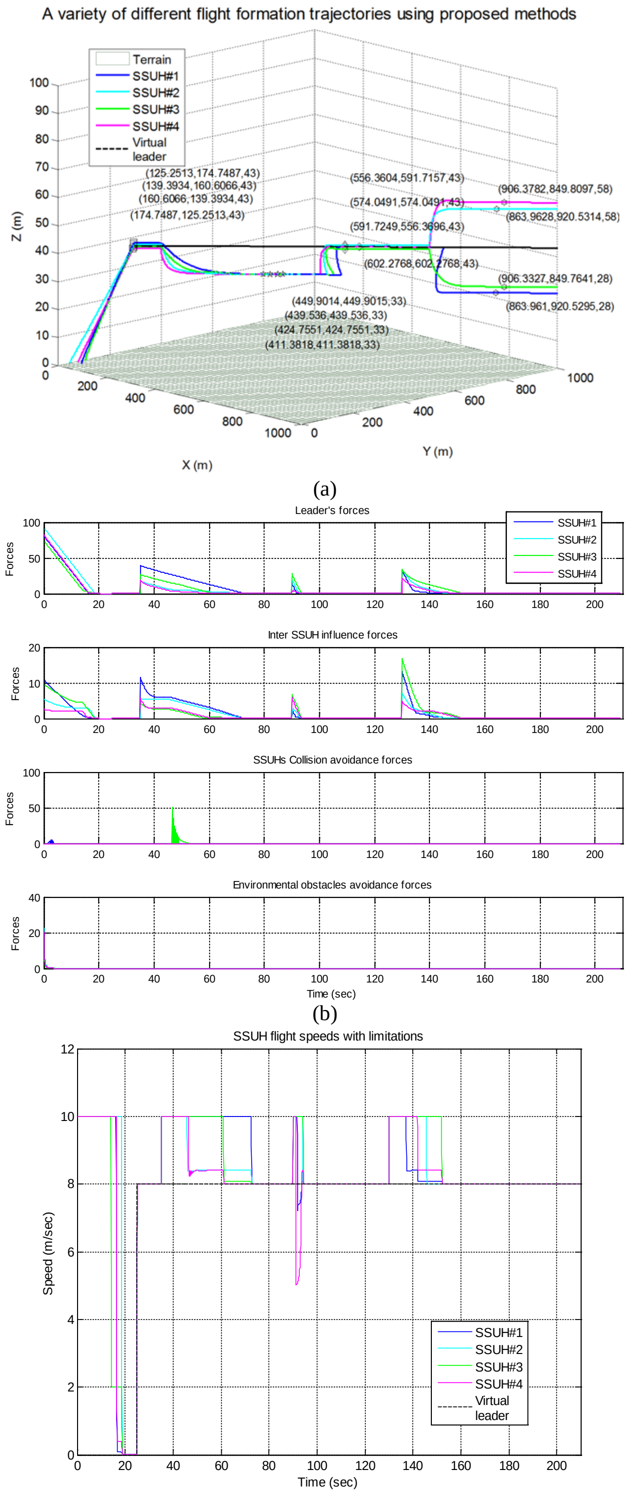
(c)

(d)

(e)

Fig. 10. The simulated formation trajectories of all the four SSUHs and the leader using the FBFN-based trajectory tracking controllers.

which means the position errors between the actual and demand (desired) flight trajectories. As can be seen in Fig. 10, there are significantly large position errors in the start-up phase due to the different initial flight positions, the second peaks of position errors occur in the beginning of formation flights owing to trajectories following, and, finally, these errors are then reduced within 5 m after they fly together in formation. The results indicate that the proposed distributed consensus MPF formation control system is effective in controlling the follower SSUHs to achieve real formation.



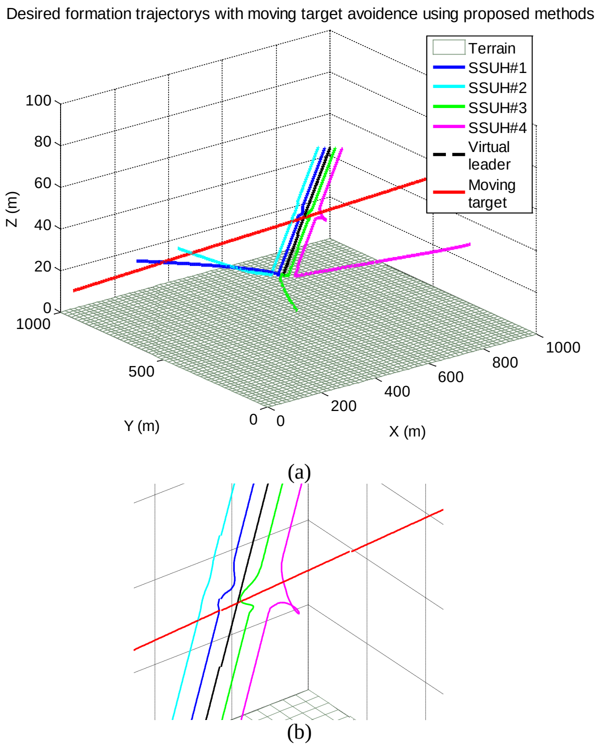
(a)

(b)

(c)

Fig. 11. The simulated formation change results of all the four SSUHs and the virtual leader using the proposed formation and control method during the formation change period; (a) the three-dimensional trajectories for the formation changes; (b) the four forces; (c) time evolutions of velocities of the four SSUHs.

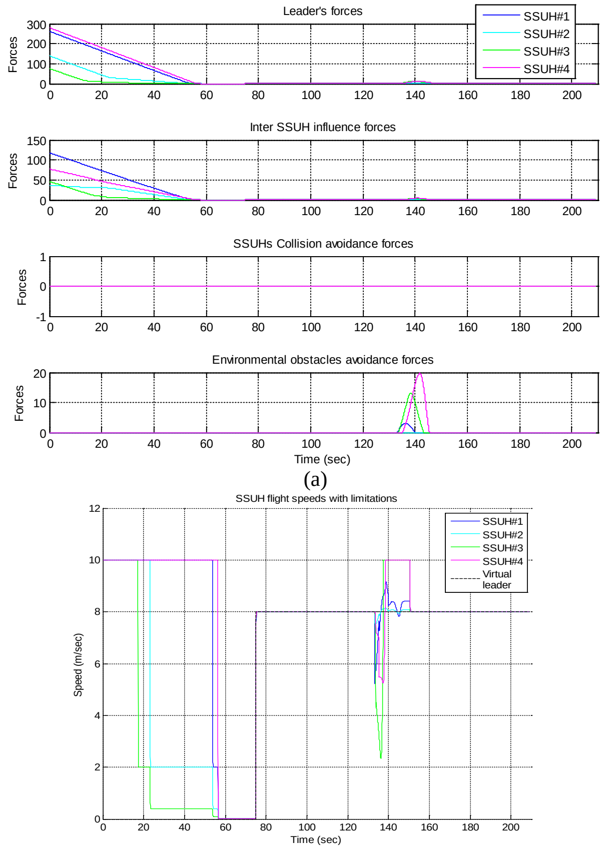
The third simulation is conducted to investigate the formation change ability of the proposed formation control system. This simulation considers the scenario that the four SSUHs start with four different initial locations and meet each other at some fixed point via hovering control from the beginning to the 25*th* second, fly together in parallel from the 25*th* second to the 70th second, change to a vertical line formation after the 70*th* second, have a triangle formation after the 100*th* second, and finally alter to a square formation after the 165*th* second. Given initial positions of the four following helicopters as (59, 32, 1), (30, 15, 1), (35, 70, 1), and (40, 38, 1) (unit:m), respectively. Fig. 11 depicts the simulated results using the proposed formation and control approach. The result in Fig. 11(b) shows that the leading and mutual forces are generated and the collision-avoidance forces are irregularly created at the moments of formation changes in order to achieve new formations and avoid any collisions among the SSUHs, and no avoidance-avoidance forces hold owing to flat terrain. We observe in Fig. 11(c) that the four SSUHs use the maximum speeds to reach the desired hovering location and stop there by the 25th second, and obvious speed changes occur shortly after the beginning of other formation changes and all the SSUH speeds keeps constant at 8 m/sec after the steady-state formation flights. The results in Fig.11 reveal that the proposed formation control method is powerful in flying all the SSUHs to carry out all the formation patterns.



(a)

(b)

Fig. 12. The simulated results of the dynamic obstacle case; (a) the 3D trajectories of the four SSUHs; (b) the three-dimensional trajectories of all the four SSUHs and the virtual leader in which the detailed trajectories are shown when the dynamic obstacle approaches the four SSUHs;



(a)

(b)

Fig. 13. The simulated results of the dynamic obstacle case; (a) the four forces of all the four SSUHs; (b) time responses of the velocities of the four SSUHs.

The fourth simulation is carried out to explore the ability of dynamic obstacle avoidance of the proposed formation control system. The initial positions of the four following helicopters are the same to those in the third simulation scenario. Fig. 12 and Fig. 13 displays the simulation results for the dynamic obstacle case, where the detailed trajectories are shown when the dynamic obstacle approaches the four SSUHs. As illustrated in Fig. 12(b), all the follower SSUHs give rise to the evident obstacle avoidance forces for the dynamic flying object, thereby preventing them from any possible collisions. As Fig. 13(b) shows, the four flying SSUHs yield remarkable speed variations in order to quickly avoid the dynamic flight object, and again maintain at 8 m/sec after the crossover. The results in Fig. 12 and Fig. 13 reveal that the proposed formation control method is capable of flying all the SSUHs to avoid dynamic obstacles.

VI. CONCLUSIONS

This paper has presented a distributed and decentralized formation control system with a novel formation trajectory generation method designed via consensus algorithm, potential field (PF) and particle swarm optimization (PSO) and real-coded genetic algorithm (RGA) for a team of small-scale unmanned helicopters (SSUHs) whose low-level trajectory tracking controllers have been synthesized using fuzzy basis function networks. The formation trajectory generator using a virtual leader-follower structure has been proposed using the improved potential function including consensus concept, and its key parameters have been optimally found by using the offline PSO-RGA algorithm. Through the four simulation results, the proposed control system have been shown effective and useful in not only accomplishing that all the follower SSUHs satisfactorily track their positions and headings given by the virtual leader, but also attaining that all the followers have good abilities to achieve collision and obstacle avoidance among the helicopters and obstacles. An interesting topic for future research would be to conduct real-time experiments of the proposed method for such multi-SSUH systems.

REFERENCES

[1] L. E. Parker, “Distributed Intelligence: Overview of the Field and its Application in Multi-Robot Systems,” *Journal of Physical Agents*, vol. 2, no. 1, pp.1-14, March 2008.

[2] R. Murray, “Recent Research in Cooperative Control of Multivehicle Systems,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, pp. 571-583, 2007.

[3] Y. Kuriki and T. Namerikawa, “Formation Control of UAVs with a Fourth-order Flight Dynamics,” in *Proc. of IEEE 52nd Annual Conference on Decision and Control (CDC 13)*, pp. 6706-6711, 2013.

[4] Y. Kuriki and T. Namerikawa, “Consensus-based Cooperative Formation Control with Collision Avoidance for a Multi-UAV System,” in *Proc. of 2014 American Control Conference* (ACC), Portland, Oregon, USA, pp. 2077-2082, June 4-6, 2014.

[5] Y. Cao and W. Ren, “Optimal Linear-Consensus Algorithms: An LQR Perspective,” *IEEE T. On Systems, Man, And Cybernetics - Part B: Cybernetics*, vol. 40, no. 3, pp. 819-830, 2010.

[6] Z. Meng, W. Ren et al., “Leaderless and Leader-Following Consensus With Communication and Input Delays Under a Directed Network Topology,” *IEEE T. on Systems, Man, and Cybernetics*, vol. 41, no. 1, pp. 75-88, 2011.

[7] W. Ren et al., “Information consensus in multivehicle cooperative Control,” *IEEE Control Systems magazine*, vol. 27, no. 2, pp. 71-82, 2007.

[8] C. Ren, and C. L. Philip Chen, “Decentralized Control for Second-order Uncertain Nonlinear Multi-agent Systems Consensus Problem Based on Fuzzy Adaptive High-gain Observer,” in *Proc. of 2013 IEEE International Conference on Systems, Man and Cybernetics*, Manchester, UK, pp. 4935-4940, 2013.

[9] T. Paul, T. R. Krogstad, and J. T. Gravdah, “ UAV Formation Control Using 3D potential Field,” *in Proc. of 16th Mediterranean Conference on Control and Automation Congress Centre, Ajaccio, France*, pp. 1240-1245, June 25-27, 2008.

[10] A. Karimoddini, H.Lin, B. M. Chen, and T. T. Lee, “Developments in Hybrid Modeling and Control of Unmanned Aerial Vehicles,” in *Proc. of 2009 IEEE International Conference on Control and Automation*, Christchurch, New Zealand, pp. 228-233, December 9-11, 2009.

[11] A. Karimoddini, H. Lin, B. M. Chen, T. H. Lee, “Hybrid Formation Control of the Unmanned Aerial Vehicles”, *Mechatronics*, vol. 21, no.5, pp. 886-898, 2011.

[12] A. Karimoddini, H. Lin, B. M. Chena, T. H. Lee, “Hybrid three-dimensional formation control for unmanned helicopters,” *Automatica*, vol. 49, no. 2, pp.424-433, 2013.

[13] M. A. Lewis and K. H. Tan, “High Precision Formation Control of Mobile Robots using Virtual Structures,” *Autonomous Robots*, Vol. 4, No. 4, 387-403, 1997.

[14] T. Balch and R. C. Arkin, “Behavior-Based Formation Control for Multirobot Teams,” *IEEE Transactions on Robotics and Automation*, Vol.14, 926-939, 1998.

[15] Z. Wang and D. Gu, “Distributed Leader-Follower Flocking Control,” *Asian Journal of Control*, Vol. 11,No. 4, 396-406, 2009.

[16] Z. Lin, M. Broucke, and B. Francis, “Local Control Strategies for Groups of Mobile Autonomous Agents,” *IEEE Transactions on Automatic Control*, Vol. 49, No. 4, 622-629, 2004.

[17] B.D.O. Anderson, Y. Changbin, B. Fidan, and J. M. Hendrickx, “Rigid Graph Control Architectures for Autonomous Formations,” *IEEE Control Systems Magazine*, Vol. 28, No. 6, 48-63, 2008.

[18] F. Fahimi, “Sliding-Mode Formation Control for Underactuated Surface Vessels,” *IEEE Transactions on Robotics*, Vol. 23, No. 3, 617-622, 2007.

[19] H. Yamaguchi, T. Arai, and G. Beni, “A Distributed Control Scheme for Multiple Robotic Vehicles to Make Group Formations,” *Robot. Auton. Syst.*, Vol. 36, 125–147, 2001.

[20] J. P. Desai, J. P. Ostrowski, and V. Kumar, “Modeling and Control of Formations of Nonholonomic Mobile Robots,” *IEEE Transactions on Robotics and Automation*, Vol. 17, No. 6, 905-908, 2001.

[21] H. Duan, Q. Luo ,G. Ma ,Y. Shi, “Hybrid Particle Swarm Optimization and Genetic Algorithm for Multi-UAV Formation Reconfiguration, ” *IEEE Computational Intelligent Magazine*, pp16-27,August 2013.

[22] J. Y. Wu, “Real-coded Genetic Algorithm-based Particle Swarm Optimization Method for Unconstrained Optimization Problems,” *Proc. of Intern. Conf. on Electronics and Information Engineering ICEIE*, V1-194-V1-198, 2010.

[23] M.W. Iruthayarajan, S. Baskar , “Optimization Of PID Parameters Using Genetic Algorithm and Particle Swarm Optimization,” *IET-UK International Conference on Information and Communication Technology in Electrical Science(ICTES 2007)*, Dr.M.G.R. University,Chennai,Tamil Nadu,India. Dec. 20-22. pp.81-86,2007.

[24] H.C Huang, S. C Lin, “Hybrid GA-PSO Algorithm for Inverse Kinematics of 7-DOF Robot,” *Proc. of 2011 International Conference on Service and Interactive Robots National Chung Hsing University, Taichung, Taiwan, Nov.25-27,2011*.

[25] C. C. Tsai, K. I. Tsai, S. C. Hsu, “Two-Loop PID Control Using PSO-RGA Algorithm for Solar Heat Pumps,” *Proc. of 2014 International Conference on Machine Learning and Cybernetics*, Lanzhou, China, 13-16 July, 2014.

[26] C. C Tsai, Z. C Wang, C. T. Lee and Y. Y. Li, “Intelligent Adaptive Trajectory Tracking Control for an Autonomous Small-Scale Helicopter Using Fuzzy Basis Function Networks,” *Asian Journal of Control* , Article first published online: 13 APR 2014, DOI: 10.1002/asjc.881.

[27] V. Gavrilets, B. Mettler, and E. Feron, “Nonlinear model for a small-size acrobatic helicopter,” *Proc. AIAA Guid. Navig. Control Conf.*, AIAA-2001–4333 (2001).

[28] F. L. Lewis, K. Hengster-Movric, A. Das, and H. Zhang, Cooperative control of multi-agent systems. Springer, 2014.

[29] C. Chen, C.-E Zen, and T. Du, “Fuzzy observed-based adaptive consensus tracking control for second-order multi-agent systems with heterogeneous nonlinear dynamics,” to appear in *IEEE Transactions on Fuzzy Systems*, 2015. DOI 10.1109/TFUZZ.2015.2486817.

[30] C. T. Lee, “3D global path planner based on genetic algorithm and elastic band technique for autonomous helicopters,” in *Proc. of 2011 International Conference on Service and Interactive Robots* , National Chung Hsing University, Taichung, Taiwan, Nov.25-27, 2011.

Ching-Chih Tsai received the Diplomat in Electrical Engineering from National Taipei Institute of Technology, Taipei, Taiwan, ROC, the MS degree in Control Engineering from National Chiao Tung University, Hsinchu, Taiwan, ROC and the Ph.D degree in Electrical Engineering from Northwestern University, Evanston, IL, USA, in 1981, 1986 and 1991, respectively. Currently, he is currently a Distinguished Professor in the Department of Electrical Engineering, National Chung-Hsing University, Taichung, Taiwan, where he served the Chairman in the Department of Electrical Engineering from 2012 to 2014. He is a Fellow of IEEE, IET and CACS.

Dr. Tsai served as the Chair, Taipei Chapter, IEEE Control Systems Society, from 2000 to 2003, and the Chair, Taipei Chapter, IEEE Robotics and Automation Society from 2005 to 2006. In 2007, he was the program chair of 2007 CACS international automatic conference sponsored by Taipei chapter, IEEE control systems society. In 2010, he served as the program co-chair of SICE 2010 annual conference in Taiwan, which was technically sponsored by IEEE CSS; in 2011, he served as the General Chair, 2011 International conference on service and interactive robotics; in 2012, he has served as the General Chair, 2012 International conference on Fuzzy Theory and Its Applications, the General Chair, 2012-2015 CACS International Automatic Control Conferences, and the General Chair, 2016-2017 International Conference on Advanced Robotics and Intelligent Systems. Dr. Tsai served the two-term President, Chinese Institute of Engineers in Central Taiwan, Taiwan from 2007 to 2011, and two-term President of Chinese Automatic Control Society from 2012 to 2015. Since 2008, he has been the Executive Directors in Boards of Government of thee professional associations, including Robotic Society of Taiwan, Taiwan Fuzzy Systems Association, and Taiwan Systems Association. He has served as the Chair, Taichung Chapter, IEEE Systems, Man, and Cybernetics Society since 2009, the Chair of IEEE SMC Technical Committee on intelligent learning in control systems since 2009, the President of Robotics Society of Taiwan since 2016, the steering committee of Asian Control Association since 2014, a BOG member of IEEE Nanotechnology council since 2012, the Vice President of International Fuzzy Systems Association since 2015, and a BOG member of the IEEE SMCS since 2017.

Dr. Tsai has published more than 500 technical papers, and seven patents in the fields of control theory, systems technology and applications. Dr. Tsai is respectively the recipients of the Third Society Prize Paper Award from IEEE Industry Application Society in 1998, the Outstanding Automatic Control Engineering Award in 2008 from Chinese Automatic Control Society (CACS), and the Outstanding Engineering Professor Award in 2009 from the Chinese Institute of Engineers in 2009, the IEEE Most Active SMC Technical Committee (TC) Award in 2012 from IEEE SMC Society, the Outstanding Electrical Engineering Professor Award from the Chinese Institute of Electrical Engineering in 2014, Outstanding Industry Contribution Award from Taiwan Systems Association in 2016, the best paper award in the International Journal of Fuzzy Systems in 2017, and many best paper awards from many international conferences technically supported by IEEE. His current interests include advanced nonlinear control methods, deep model predictive control, fuzzy control, neural-network control, advanced mobile robotics, intelligent service robotics, intelligent mechatronics, intelligent learning control methods with their applications to industrial processes and intelligent machinery.

Ching-Fu Hsu received the B.S. degree in Electronic Engineering from Chienkuo Technology University in 2001 and M.S. degree in Department of Computer Science and Information Engineering from Chaoyang University of Technology in 2003, respectively. From 2003 to 2015, he served a project leader in Taiwan Wintek Corp. Since 2015, he has been with Taiwan ASUS Computer Corp. Currently, he is working toward his Ph.D degree at Department of Electrical Engineering, National Chung Hsing University. His current interests include image processing and driving technology and their applications to touch/display modules.

Zen-Chung Wang received his M.S. degree from Chung Cheng Institute of Technology, Taoyuan, Taiwan. Currently, he is a Ph.D. candidate in the Department of Electrical Engineering, National Chung-Hsing University, Taichung, Taiwan. Mr. Wang also works as an engineer for Chung-Shan Institute of Science & Technology, Taiwan. His research interest is in electronic control system design, especially in training simulators.

Feng-Chun Tai received the B.S., M.S. and Ph.D. degrees in Department of Electrical Engineering from National Chung Hsing University, Taichung, Taiwan, ROC. in 2007, 2010 and 2018, respectively. His current research interests include mobile robots, intelligent control, navigation system and their applications to industrial processes and machines.

Parameter Tuning of Optimal Motion Control for Mecanum Mobile Robots Using Bacterial Foraging Optimization

Hsu-Chih Huang, *Member, IEEE* and Shao-Kang Lin

Abstract- **This paper presents an evolutionary parameter tuning method of optimal motion control for Mecanum mobile robots using bacterial foraging optimization (BFO). A metaheuristic BFO is employed to address the parameter tuning problemof motion control for four-wheeled Mecanum mobile robots. The redundant kinematics problem is resolved by means of the pseudoinverse control method. By considering the derived vehicle kinematics and Lyapunov stability theory, an optimal motion control law is synthesized using the natured-inspired BFO paradigm. This intelligent motion control scheme surpasses the traditional control approaches because the control parameters are optimized. Simulation results illustrate the merit and effectiveness of the proposed BFO-based optimal controller for Mecanum mobile vehicles.**

***Index Terms*—BFO, metaheuristic, mobile robot, optimization, intelligent control.**

I. Introduction

IN robotics research, parameter tuning of motion control is an important topic in designing optimal controllers. The control parameters in the motion controllers are properly determined to obtain better performance. This tuning problem has attracted much attention in both academic and industry. It is regarded as an optimization problem and has been investigated by several studies using various approaches. Overall, these works are computationally intensive and may fall into local optimum for designing optimal controllers [1-5].

Swarm intelligence is a new branch used for dealing with optimization problems. This computing paradigm is a category of computational intelligences. There are some popular algorithms widely applied to resolve multi-objective optimization problems in various application domains [6-7], such as GA (Genetic Algorithm), PSO (Particle Swarm Optimization), ACO (Ant Colony Optimization) and BA (Bat Algorithm) [6-10]. They have become powerful tools to address many real-world engineering problems by exploiting their strong optimization ability.

BFO introduced by Passino is relatively new algorithm to the family of swarm intelligence [11-12]. This population-based algorithm is inspired by the group foraging behavior of Escherichia coli (E. coli) bacteria. BFO has already drawn the attention of researchers because of its efficiency in solving complex optimization problems based on the chemotaxis, swarming, reproduction and elimination-dispersal steps. It has been proven superior to the traditional swarm intelligences by taking the advantages of simplicity and ease of implementation [13-14]. However, there has no attempt to develop BFO-based evolutionary controller for Mecanum mobile robots.

Four-wheeled mobile robots with Mecanum omnidirectional wheels have received growing attention in the mobile robotics research [15-16]. Although such robots outperform the conventional two-wheeled differential robots and three-wheeled holonomic mobile robots, there exists a redundant control problem in the robot system. Several studies have been proposed to investigate the modeling and control of the redundant mobile robots [17-20]. Overall, these works did not cope with the BFO-based parameter tuning of optimal motion control.

Evolutionary control is a challenging methodology for designing optimal controllers. This approach combines the soft computing and hard computing to develop optimal controllers. Taking the advantages of the metaheurictics and robot modeling, this control scheme has become popular in robotics research. To date, some evolutionary control methods for mobile robots have been presented [21-24]. As the authors' best understanding, the BFO parameter tuning of optimal motion control for four-wheeled Mecanum autonomous robots remains open.

The objective of this research is to develop a metaheuristic BFO-based parameter tuning method, and apply this pragmatic computing to optimal motion control of four-wheeled Mecanum vehicles. The remaining of this paper is organized as follows. In Section II, the BFO parameter tuning and vehicle modeling are introduced. Section III elaborates the application to optimal motion control of four-wheeled Mecanum vehicles to achieve trajectory tracking. Section IV conducts simulation results to show the effectiveness of the proposed methods. Section V concludes this paper.

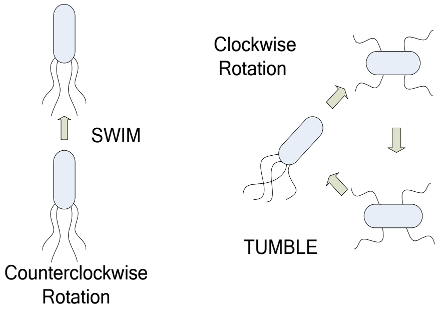


Fig. 1. Swim and tumble of a bacterium.

Hsu-Chih Huang and Shao-Kang Lin are with the Department of Electrical Engineering, National Ilan University, Yilan, Taiwan.
(Corresponding author Hsu-Chih Huang, email: hchuang@niu.edu.tw)

II. BFO Parameter Tuning and Vehicle Modeling

A. Classical BFO

1) Foraging behavior of the real bacteria

The locomotion of the real bacteria during foraging is achieved by a set of tensile flagella. These flagella help an E. coli bacterium to tumble or swim in the environment. When they rotate the flagella in the clockwise direction, the bacterium tumbles. On the other hand, moving the flagella in the counterclockwise direction helps the bacterium to swim. The bacteria move towards a nutrient gradient and avoid noxious environment based on these two operations. Fig. 1 presents the swim and tumble of a bacterium. In BFO, the foraging strategy consists of four principal mechanisms, including chemotaxis, swarming, reproduction, and elimination-dispersal.

2) Chemotaxis

The movement of E. Coli bacteria towards the nutrient-rich area is simulated by a biological activity called chemotaxis. This operation simulates the movement of an E. coli through swimming and tumbling. Suppose $\theta^i(j,k,l)$ represents the position of ith bacterium at j^{th} chemotaxis, kth reproductive and lth elimination-dispersal step. $C(i)$ is the step size by the tumble operation. The computational chemotaxis of the bacterium is expressed by

$$\begin{aligned}\theta^i\left(j+1,k,l\right) &= \theta^i\left(j,k,l\right)+C\left(i\right)\varphi\left(j\right)\\ &= \theta^i\left(j,k,l\right)+C\left(i\right)\frac{\Delta\left(i\right)}{\sqrt{\Delta^T\left(i\right)\Delta\left(i\right)}}\end{aligned} \tag{1}$$

where $\Delta(i)$ is a random vector. Note that N_C is the number of chemotactic step and N_S is the swimming length.

3) Swarming

In BFO, the swarming operation makes the bacteria come together into groups with high bacterial density to search the food. This operation is expressed by

$$\begin{aligned}J_{cc}\left(\theta,P\left(j,k,l\right)\right) &= \sum_{i=1}^{S} J_{cc}\left(\theta,\theta^i\left(j,k,l\right)\right)\\ &= \sum_{i=1}^{S}\left[-d_{attrac\tan t}\exp\left(-w_{attrac\tan t}\sum_{m=1}^{p}\left(\theta_m-\theta_m^i\right)^2\right)\right]\\ &+\sum_{i=1}^{S}\left[h_{repellant}\exp\left(-w_{attrac\tan t}\sum_{m=1}^{p}\left(\theta_m-\theta_m^i\right)^2\right)\right]\end{aligned} \tag{2}$$

where $P(j,k,l)=\left\{\theta^i(j,k,l)\middle|i=1,2,...,S\right\}$ represent the position of each member in the population of the S bacteria at the j^{th} chemotactic step, k^{th} reproduction step, and l^{th} elimination-dispersal event. $J_{cc}(\theta,P(j,k,l))$ is the cost function value to be added to the objective function $J(\theta)$, $\theta\in R^p$ of the optimization problem. S is the total number of the bacteria and p is the number of variables to be optimized. $\theta=[\theta_1\,\theta_2\ldots\theta_p]^T$ is a point on the search space. $d_{attractant}$ is the depth of the attractant released by the cell and $w_{attractant}$ is a measure of the width of the attractant signal. $h_{repellant}$ is the height of the repellant effect magnitude and $w_{repellant}$ is a measure of the width of the repellant. The cell-to-cell signaling in E. coli swarming behavior is expressed by

$$J(i,j+1,k,l)=J(i,j,k,l)+J_{cc}(\theta^i,(j+1,k,l),P(j+1,k,l)) \tag{3}$$

4) Reproduction

In BFO algorithm, the reproduction operation keeps the swarm constant. The half least healthy bacteria eventually die and the better half reproduces to generate the next generation. N_{re} is defined as the reproduction step in evolutionary BFO computing.

5) Elimination and dispersal

In the evolution process, gradual or sudden changes in the environment may occur due to various reasons, including rise of temperature and consumption of nutrients. This phenomenon may kill a group of bacteria in a region. BFO algorithm applies elimination and dispersal process to simulate the behavior, thereby preventing BFO be trapped at local optimum. In BFO algorithm, P_{ed} is the elimination-dispersal probability and N_{ed} is the number of elimination-dispersal events. The following summarizes the BFO using the evolutionary processes.

Step 1 Initialize parameters p, S, N_c, N_s, N_{re}, N_{ed}, P_{ed}，$C(i)$ $(i=1,2,\ldots,S)$, and θ^i. A BFO individual is defined as the sequence of parameter tuning problem.

Step 2 Elimination-dispersal loop：$l=l+1$

Step 3 Reproduction loop：$k=k+1$

Step 4 Chemotaxis loop：$j=j+1$

1) For i=1,2,⋯,S, take a chemotactic step for bacterium i.

2) Compute fitness function $J(i,j,k,l)$

3) Tumble process: Generate a random vector $\Delta(i)\in R^p$.

4) Move using (1.2), this results in a step of size $C(i)$ in the direction of the tumble for bacterium i .

5) Compute $J(i,j+1,k,l)$.

6) Swim

7) Go to next bacterium $i+1$

Step 5 If $j<Nc$, go to Step 4. In this case, continue chemotaxis since the life of the bacteria is not over.

Step 6 Reproduction:

1) Sort bacteria and chemotactic parameters C(i) in order of ascending cost Jhealth.

2) The Sr bacteria with the highest Jhealth values die and the remaining Sr=S/2 bacteria with the best values split.

Step 7 If $k<N_{re}$, go to Step 3

Step 8 Elimination-dispersal. For $i=1,2,\ldots S$, with probability P_{ed}, eliminate and disperse each bacterium. If $l<N_{ed}$, go to Step 2, otherwise end this algorithm.

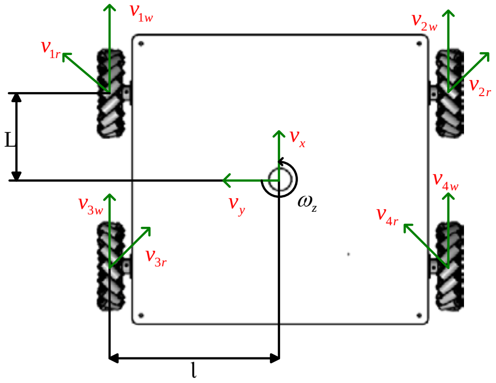


Fig. 2. The kinematic model of four-wheeled Mecanum omnidirectional mobile robots.

B. *Vehicle modeling*

This subsection is devoted to introducing the kinematic model of four-wheeled Mecanum omnidirectional mobile robots with four independent driving wheels as shown in Fig. 2. *ω$_z$* is the moving angle of the robot. *L* and *l* are the distance between the centers of each Mecanum wheels to the robot platform of the geometric center, $v_{iw}(i=1,2,3,4)\in R$ is the Mecanum wheels velocity vector corresponding to each Mecanum wheels. $v_{ir}(i=1,2,3,4)\in R$ is the tangential velocity vector for each free roller contact to the ground.

Since the four-wheeled Mecanum omnidirectional mobile robot is structure symmetry, the geometric center and the center of mass of the robot are overlapped. The kinematic model for the four-wheeled Mecanum omnidirectional mobile robot is derived as follows:

$$\begin{array}{ll} v_{1x}=v_{1w}+\dfrac{v_{1r}}{\sqrt{2}},v_{1y=}\dfrac{v_{1r}}{\sqrt{2}} & v_{2x}=v_{2w}+\dfrac{v_{2r}}{\sqrt{2}},v_{2y=}\dfrac{v_{2r}}{\sqrt{2}} \\ v_{3x}=v_{3w}+\dfrac{v_{3r}}{\sqrt{2}},v_{3y=}\dfrac{v_{3r}}{\sqrt{2}} & v_{4x}=v_{4w}+\dfrac{v_{4r}}{\sqrt{2}},v_{4y=}\dfrac{v_{2r}}{\sqrt{2}} \end{array} \tag{4}$$

$$\begin{array}{ll} v_{1x}=v_x-l\omega_z & v_{1r}=v_y+L\omega_z \\ v_{2x}=v_x-l\omega_z & v_{2r}=v_y+L\omega_z \\ v_{3x}=v_x-l\omega_z & v_{3r}=v_y+L\omega_z \\ v_{4x}=v_x-l\omega_z & v_{4r}=v_y+L\omega_z \end{array} \tag{5}$$

v_{ix},v_{iy}, and $\omega_z\in R$ represent the velocity and angular velocity of the vehicle *x* and *y* for wheel *i*. Combining (1) and (2), one obtains the following kinematic model of the Mecanum mobile robots

$$\begin{bmatrix} v_{1w} \\ v_{2w} \\ v_{3w} \\ v_{4w} \end{bmatrix} = J\begin{bmatrix} v_x \\ v_y \\ \omega_z \end{bmatrix} = J\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \tag{6}$$

where $\left[x\ y\ \theta\right]^T$ is the position and orientation of the mobile robot and

$$J=\begin{bmatrix} 1 & -1 & -(l+L) \\ 1 & 1 & (l+L) \\ 1 & 1 & -(l+L) \\ 1 & -1 & (l+L) \end{bmatrix} \tag{7}$$

Notice that *J* is a singular matrix, its pseudo-inverse matrix *J*$^{-1}$ can be found by

$$J^{-1}=\frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ -\dfrac{1}{L+l} & \dfrac{1}{L+l} & -\dfrac{1}{L+l} & \dfrac{1}{L+l} \end{bmatrix} \tag{8}$$

where $J^{-1}J=I_3$.

III. BFO-BASED OPTIMAL MOTION CONTROL OF MECANUM MOBILE ROBOTS

A. *Design of motion control*

Based on the kinematic model in (3), the goal of the control law is to design the velocity vector $\left[v_{1w}\quad v_{2w}\quad v_{3w}\quad v_{4w}\right]^T$ to steer the four-wheeled Mecanum omnidirectional mobile robot from any starting pose $\left[x_0\quad y_0\quad \theta_0\right]^T$ to any desired pose $\left[x_d(t)\quad y_d(t)\quad \theta_d(t)\right]^T$. The current pose is $\left[x\quad y\quad \theta\right]^T$. In order to design the tracking controller, one define the error vector by

$$\begin{bmatrix} x_e(t) \\ y_e(t) \\ \theta_e(t) \end{bmatrix} = \begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \end{bmatrix} - \begin{bmatrix} x_d(t) \\ y_d(t) \\ \theta_d(t) \end{bmatrix} \tag{9}$$

and

$$\begin{bmatrix} \dot{x}_e(t) \\ \dot{y}_e(t) \\ \dot{\theta}_e(t) \end{bmatrix} = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = J^{-1}\begin{bmatrix} v_{1w}(t) \\ v_{2w}(t) \\ v_{3w}(t) \\ v_{4w}(t) \end{bmatrix} \tag{10}$$

In order to find the speed control $\left[v_{1w}\quad v_{2w}\quad v_{3w}\quad v_{4w}\right]^T$ that make the closed-loop system can asymptotically stable, one propose the following control law:

$$\begin{bmatrix} v_{1w}(t) \\ v_{2w}(t) \\ v_{3w}(t) \\ v_{4w}(t) \end{bmatrix} = J\left(-K_p\begin{bmatrix} x_e(t) \\ y_e(t) \\ \theta_e(t) \end{bmatrix} - K_I\begin{bmatrix} \int_0^t x_e(\tau)d\tau \\ \int_0^t y_e(\tau)d\tau \\ \int_0^t \theta_e(\tau)d\tau \end{bmatrix} - K_D\begin{bmatrix} \dot{x}_d(t) \\ \dot{y}_d(t) \\ \dot{\theta}_d(t) \end{bmatrix} \right) \tag{11}$$

The control is a PID-like control law with three terms. K_P, K_I and K_D are gain matrices. They are symmetric and positive definite. The closed-loop error system can be found as follows:

$$\begin{bmatrix}\dot{x}_e(t)\\ \dot{y}_e(t)\\ \dot{\theta}_e(t)\end{bmatrix} = \left(-K_P\begin{bmatrix}x_e(t)\\ y_e(t)\\ \theta_e(t)\end{bmatrix} - K_I\begin{bmatrix}\int_o^t x_e(\tau)d\tau\\ \int_o^t y_e(\tau)d\tau\\ \int_o^t \theta_e(\tau)d\tau\end{bmatrix} - K_D\begin{bmatrix}\dot{x}_e(t)\\ \dot{y}_e(t)\\ \dot{\theta}_e(t)\end{bmatrix}\right) \qquad (12)$$

$$= \begin{bmatrix}\dot{x}_d(t)\\ \dot{y}_d(t)\\ \dot{\theta}_d(t)\end{bmatrix}$$

To prove the closed-loop error system as asymptotic stable, we select the radially unbounded Lyapunov function candidate as follows:

$$V = \frac{1}{2}\left[x_e(t) \quad y_e(t) \quad \theta_e(t)\right]\begin{bmatrix}x_e(t)\\ y_e(t)\\ \theta_e(t)\end{bmatrix}$$

$$+\frac{1}{2}\left[\int_o^t x_e(\tau)d\tau \quad \int_o^t y_e(\tau)d\tau \quad \int_o^t \theta_e(\tau)d\tau\right]K_I\begin{bmatrix}\int_o^t x_e(\tau)d\tau\\ \int_o^t y_e(\tau)d\tau\\ \int_o^t \theta_e(\tau)d\tau\end{bmatrix} \qquad (13)$$

$$+\frac{1}{2}\left[x_e(t) \quad y_e(t) \quad \theta_e(t)\right]K_D\begin{bmatrix}x_e(t)\\ y_e(t)\\ \theta_e(t)\end{bmatrix} > 0$$

and

$$\dot{V} = \left[x_e(t) \quad y_e(t) \quad \theta_e(t)\right]\left(-K_P\begin{bmatrix}x_e(t)\\ y_e(t)\\ \theta_e(t)\end{bmatrix}\right) < 0 \qquad (14)$$

Since $\dot{V}$ is negative definite, Lyapunov theory proves that $\left[x_e(t) \quad y_e(t) \quad \theta_e(t)\right]^T \to 0$ as $t \to \infty$. The closed-loop error system is asymptotical stable.

B. BFO-based optimal motion control

In this study, the BFO parameters are tuned via the classical genetic algorithm. The chromosome is defined as the parameter sequence. With the genetic operations, the optimal parameters are determined. This modified BFO algorithm is then employed to design an optimal motion controller using the vehicle model of Mecanum mobile robot.

The next goal of this subsection is to apply the proposed modified BFO to synthesize an evolutionary controller. More precisely, the control parameters K_P, K_I and K_D in (11) are properly determined by means of the proposed BFO computing. This control approach outperforms the trial-and error method in traditional motion controllers. The objection function of the proposed BFO is defined based on the weighted integral square error (ISE), expressed by

$$F = w_{bfo}\int_0^t \left(x_e^2(\tau) + y_e^2(\tau) + \theta_e^2(\tau)\right)d\tau \qquad (15)$$

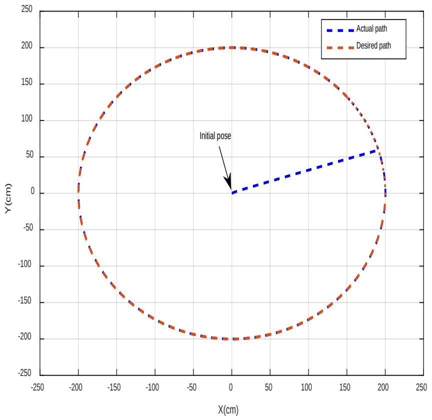


Fig. 3. Simulation result of the circular trajectory tracking.

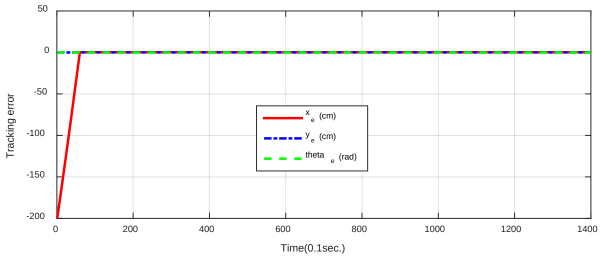


Fig. 4. Tracking errors of the circular trajectory tracking.

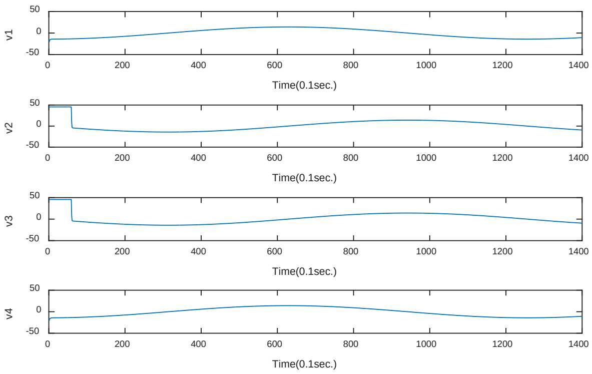


Fig. 5. Controller's output to the four omnidirectional wheels.

where w_{bfo} is a weighting factor, x_e and y_e are the position errors. θ_e is the error of vehicle's orientation of the four-wheeled Mecanum autonomous vehicles.

IV. Simulation Results and Discussions

The first simulation is conducted to examine the effectiveness of the proposed BFO-based evolutionary controller. The desired trajectory is a circular trajectory: $[x_d(t), y_d(t), \theta_d(t)]^T = [200\cos(\omega_i t)$ cm, $200\sin(\omega_i t)$ cm, π/4rad$]^T$, ω_i=0.15 rad/sec The initial pose is set at the origin. Fig. 3 presents the simulation result of circular trajectory tracking and Fig. 4 depicts the tracking errors for the four-wheeled Mecanum vehicle with BFO parameter tuning. All the errors converge to zero in 5 seconds. The controller's output to the four omnidirectional wheels is presented in Fig. 5. As shown in Figs. 3-5, the proposed BFO optimal redundant controller (8) successfully steers the vehicle to track this circle trajectory with satisfactory performance.

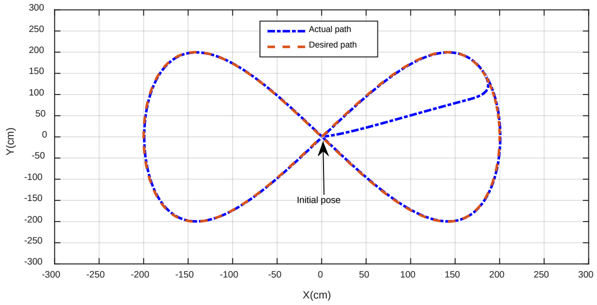


Fig. 6. Simulation result of the Lemniscate curve trajectory tracking.

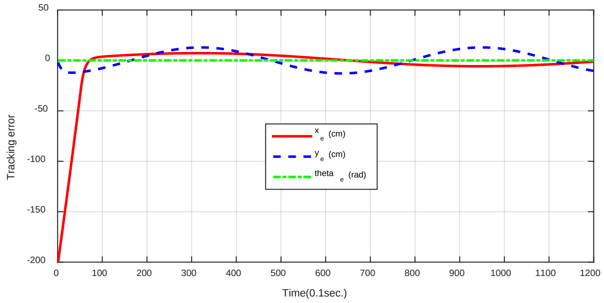


Fig. 7. Tracking errors of the Lemniscate curve trajectory tracking.

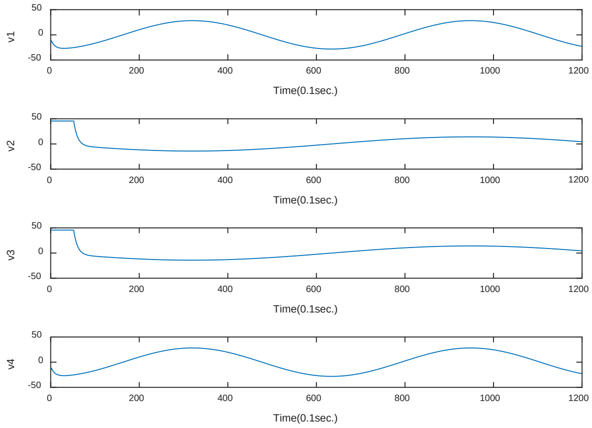


Fig. 8. Simulation result of the Lemniscate curve trajectory tracking.

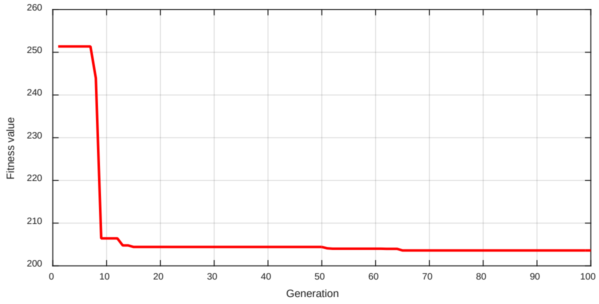


Fig. 9. Fitness value of the BFO-based controller to achieve the Lemniscate curve tracking.

This second simulation is used to investigate the tracking performance of Lemniscate curve trajectory using the proposed BFO parameter tuning method. The desired trajectory is: $[x_d(t), y_d(t), \theta_d(t)]^T = [200\cos(\omega_i t)$ cm, $200\sin(2\omega_i t)$ cm, 0rad$]^T$, ω_i=0.2 rad/sec. The initial pose is set at the origin. Fig. 6 presents the simulation result of the optimal BFO controller to track this special trajectory. The tracking errors of position and orientation are depicted in Fig. 7.

Moreover, Fig. 8 depicts the controller's output of the proposed BFO control scheme. The convergent behavior of the intelligent BFO controller is depicted in Fig. 9. These simulation results clearly indicate that the proposed BFO-based controller with parameter tuning steers the four-wheeled Mecanum vehicle to track this trajectory.

V. Conclusions

This paper has presented an evolutionary parameter tuning method of optimal motion control for Mecanum mobile robots using BFO computing paradigm. By considering the derived vehicle kinematics and Lyapunov stability theory, an optimal motion control law is synthesized using the natured-inspired BFO paradigm. The control parameters are well-tuned to obtain optimal performance by means of the BFO computing approach. Simulation results illustrate the merit and effectiveness of the proposed BFO-based optimal controller for Mecanum mobile vehicles.

References

[1] J. Jung and K. Kong, "Mechanical parameter tuning based on iterative learning mechatronics approach," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 2, pp. 906-915, 2018.

[2] C. Kwak, M. Uhm I. Yom and H. J. Eom, "Automated microwave filter tuning using curve similarity and weighted least squares," *IEEE Microwave and Wireless Components Letters*, vol. 22, no. 10, pp. 539-541, 2012.

[3] J. Liu, Z. Li, L. Guan and L. Pan, "A novel parameter-tuned stochastic resonator for binary PAM signal processing at low SNR," *IEEE Communications Letters*, vol. 18, no. 3, pp. 427-430, 2014.

[4] S. B. Lee, "Closed-loop estimation of permanent magnet synchronous motor parameters by PI controller gain tuning," *IEEE Transactions on Energy Conversion*, vol. 21, no. 4, pp. 863-870, 2006.

[5] A. S. Millan, S. S. Aphale and V. Feliu, "A fast algebraic estimator for system parameter estimation and online controller tuning-a nanopositioning application," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 6, pp. 4534-4543, 2019.

[6] P. A. S. Dayal, G. S. N Raju and S. Mishra, "Pattern synthesis using accelerated particle swarm optimization, *Modelling, Measurement and Control*, vol. 89, no. 1, pp. 58-76, 2016.

[7] H. Shen, G. Liu and H. Chandler, "Swarm intelligence based file replication and consistency maintenance in structured P2P file sharing systems," *IEEE Transactions on Computers*, vol. 64, no. 10, pp. 2953-2967, 2015.

[8] A. H. Gandomi and A. R. Kashani, "Construction cost minimization of shallow foundation using recent swarm intelligence techniques," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 1099-1106, 2018.

[9] V. Kaltsa, A. Briassouli, I. Kompatsiaris, L. J. Hadjileontiadis and M. Strintzis, "Swarm intelligence for detecting interesting events in crowded environments," *IEEE Transactions on Image Processing*, vol. 24, no. 7, pp. 2153-2166, 2015.

[10] A. Slowik and H. Kwasnicka, "Nature inspired methods and their industry applications-swarm intelligence algorithms," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 1004-1015, 2018.

[11] Q. Zhang, H. Chen, J. Luo, Y. Xu, C. Wu and C. Li, "Chaos enhanced bacterial foraging optimization for global optimization," *IEEE Access*, vol. 6, pp. 64905-64919, 2018.

[12] W. J. Tang, M. S. Li, Q. H. Wu and J. R. Saunders, "Bacterial foraging algorithm for optimal power flow in dynamic environments", *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 55, no. 8, pp. 2433-2442, 2008.

[13] L. S. Coelho, C. C. Silveira, C. A. Sierakowski and P. Alotto, "Improved bacterial foraging strategy applied to TEAM workshop benchmark problem," *IEEE Transactions on Magnetics*, vol. 46, no. 8, pp.2903-2906,

2010.

[14] M. A. Awadallah and B. Venkatesh, “Bacterial foraging algorithm guided by particle swarm optimization for parameter identification of photovoltaic modules,” *Canadian Journal of Electrical and Computer Engineering*, vol. 39, no. 2, pp. 150-157, 2016.

[15] M. Y. Naing, A. S. Oo, I. Nilkhamhang and T. Than, “Development of computer vision-based movement controlling in Mecanum wheel robotic car,” *2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP)*, pp. 45-48, 2019.

[16] E. Malayjerdi, H. Kalani and M. Malayjerdi,“Self-tuning fuzzy PID control of a four-Mecanum wheel omni-directional mobile platform,” *Iranian Conference on Electrical Engineering (ICEE)*, pp. 816-820, 2018.

[17] J. G. Shuai and J. F. Guo,“Mechatronic model and simulation of four-wheel Mecanum vehicle driven by BLDC,” *2011 International Conference on Consumer Electronics, Communications and Networks (CECNet)*, pp. 2317-2320, 2011.

[18] R. Wen and M. Tong, “Mecanum wheels with Astar algorithm and fuzzy PID algorithm based on genetic algorithm,” *2017 International Conference on Robotics and Automation Sciences (ICRAS)*, pp. 114-118, 2017.

[19] M. Wada and H. Asada, “Design and control of a variable footprint mechanism for holonomic omnidirectional vehicles and its application to wheelchairs,” *IEEE Transactions on Robotics and Automation*, vol. 15, no. 6, pp. 978-989, 1999.

[20] R. Siegwart, I. R. Nourbakhsh and D. Scaramuzza, “Introduction to autonomous mobile robots,” *2nd Edition, MIT Press*, London, 2011.

[21] J. A. Cabrera, J. J. Castillo, E. Carabias and A. Ortiz, “Evolutionary optimization of a motorcycle traction control system based on fuzzy logic,” *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 5, pp. 1594-1607, 2015.

[22] J. A. Cabrera, J. J. Castillo, E. Carabias and A. Ortiz, “Implementation of evolutionary fuzzy PID speed controller for PM synchronous motor,” *IEEE Transactions on Industrial Informatics*, vol. 11, no. 2, pp. 540-547, 2015.

[23] M. Duarte, J. Gomes, S. M. Oliveira and A. L. Christensen, “Evolution of repertoire-based control for robots with complex locomotor systems,” *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 314-328, 2018.

[24] A. Yanping and Q. Ping, “Air-conditioning energy-saving control strategy at subway station based on MAS evolutionary algorithm,” *2015 8th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, pp. 122-125, 2015.

Hsu-Chih Huang received the M.S. degree in Institute of Biomedical Engineering from National Cheng-Kung University, Tainan, Taiwan, and Ph.D in Electrical Engineering from National Chung-Hsing University, Taichung, Taiwan, in 1999, and 2009, respectively.

He is currently a Professor in the Department of Electrical Engineering, National Ilan University, Yilan, Taiwan. His current research interests include intelligent control, mobile robots, embedded systems, SoPC and nonlinear control.

Shao-Kang Lin is currently working toward his M.S degree in the Department of Electrical Engineering, National Ilan University.

His current research interests include intelligent control, mobile robots, embedded systems and SoPC.

Sum of Squares-based Control of an Underactuated Robot Using Polynomial Fuzzy Model

Gwo-Ruey Yu, *Member, IEEE*, Yu-Shan Chiu, and Yu-Chia Huang

Abstract- **This paper proposed the design of an underactuated robot controller based on polynomial fuzzy systems. First, we obtain the nonlinear state equations by the properties of the underactuated robot systems. Second, the polynomial fuzzy model can be constructed by the nonlinear state equations. We apply the strategy of parallel distributed compensation to design the nonlinear controller. To guarantee the stability of the underactuated robot system, the constraint conditions are derived using the polynomial Lyapunov function. Furthermore, the stability conditions are represented in terms of sum of squares of the polynomial fuzzy model. In order to improve the performance of control system, we use the SOSTOOL to design the polynomial fuzzy controller.**

Index Terms—Underactuated robot, polynomial fuzzy systems, sum-of-squares.

I. Introduction

THE Takagi-Sugeno (T-S) fuzzy control approach includes T-S fuzzy model, parallel distributed compensation (PDC) controller, and quadratic Lyapunov function-based stability analysis [1]. The T-S fuzzy model can exactly describe a nonlinear system with IF-THEN rules by the sector nonlinearity concept, and the PDC controller is constructed based on the same membership with T-S fuzzy model. Finally, the stability analysis ensures the controlled system is stable. The T-S fuzzy control has been widely applied in many engineering problems [2]-[7].

In 2009, the polynomial fuzzy control with sum-of-squares (SOS) approach was proposed [8]. The polynomial fuzzy model and controller are more general and effective presentation than T-S fuzzy model and PDC controller. The stability conditions of the polynomial fuzzy control are presented in tern of SOS. Solving the stability conditions via SOSTOOL [9], the controller is carried out. Recently, the polynomial fuzzy control has received great attentions [10]-[16].

In this paper, the controller of a Pendubot system is designed by polynomial fuzzy control. First the fuzzy model of the Pendubot is built by linearization and fuzzy membership functions. By the SOS-based stability conditions, the feedback gains are obtained. There are some experiments about regulation, tracking, disturbance rejection, and uncertainty, respectively. In order to achieve the performance requirement, the pole placement method is used to seek the control gains, and then the stability of the fuzzy system can be ensured by Theorem 1 proposed in this paper.

This paper is organized as follows. Section II includes the introduction of the polynomial fuzzy model and controller, a proposition, a lemma, and a theorem. Section III describes the Pendubot system and its fuzzy model. Section IV presents the results. Section V gives conclusions.

II. Polynomial Fuzzy Control

A. Polynomial Fuzzy Model

Using polynomial fuzzy modeling, a nonlinear equation can be represented by several subsystems. The i th subsystem is defined as:

Rule i :

$$\text{IF } z_1(t) \text{ is } M_{i1} \text{ AND} \cdots \text{AND } z_p(t) \text{ is } M_{ip},$$

$$\text{THEN } \dot{x}(t) = \boldsymbol{A}_i(x(t))\hat{x}(x(t)) + \boldsymbol{B}_i(x(t))u(t),\ i = 1,2,\cdots,r. \quad (1)$$

Therefore, the total output of the polynomial fuzzy system can be described as:

$$\dot{x}(t) = \sum_{i}^{r} h_i(z(t))\left\{\boldsymbol{A}_i(x(t))\hat{x}(x(t)) + \boldsymbol{B}_i(x(t))u(t)\right\}. \quad (2)$$

where $\sum_{i=1}^{r} h_i(z(t)) = 1$, $h_i(z(t)) \in [0 \quad 1]$ for all i ,

$$h_i(z(t)) = \frac{\prod_{j=1}^{p} \mu_{M_{ij}}(z_j(t))}{\sum_{k=1}^{r}\left(\prod_{j=1}^{p} \mu_{M_{kj}}(z_j(t))\right)} \quad (3)$$

where r is the number of the rules; p is the number of the premise variables; $z_1(t),\cdots,z_p(t)$ are the premise variables and $z(t) = [z_1(t),\cdots,z_p(t)]$; $x(t) \in R^n$ is the state vector, and $\hat{x}(x(t)) \in R^N$ is a vector whose entries are defined as $x_1^{v_1} x_2^{v_2} \cdots x_n^{v_n}$, where v_1, v_2, …, v_n are nonnegative integers;

Gwo-Ruey Yu is with Department of Electrical Engineering, National Chung Cheng University, Chia-Yi, Taiwan (e-mail: ieewoyu@ccu.edu.tw).
Yu-Shan Chiu is with Graduate Institute of Opto-Mechatronics, National Chung Cheng University, Chia-Yi, Taiwan.
Yu-Chia Huang is with Department of Electrical Engineering, National Taiwan Ocean University, Keelung, Taiwan.
Hsu-Chih Huang and Shao-Kang Lin are with the Department of Electrical Engineering, National Ilan University, Yilan, Taiwan.
(Corresponding author Hsu-Chih Huang, email: hchuang@niu.edu.tw)

$u(t) \in R^m$ is the input vector; $\boldsymbol{A}_i(x(t)) \in R^{n\times N}$, $\boldsymbol{B}_i(x(t)) \in R^{n\times m}$ are polynomial matrices; M_{ip} is the fuzzy set; and $\mu_{M_{ij}}(z_j(t))$ is the grade of membership of $z_j(t)$ in M_{ij}.

A. Polynomial Fuzzy Controller

The polynomial fuzzy controller shares the same membership with polynomial fuzzy model. Every rule is presented as follows.

Rule i:

$$\begin{aligned} &\text{IF } z_1(t) \text{ is } M_{i1} \text{ AND} \cdots \text{AND } z_p(t) \text{ is } M_{ip}, \\ &\text{THEN } u(t) = -\boldsymbol{F}_i(x(t))\hat{x}(x(t)), i = 1,2,\cdots,r. \end{aligned} \tag{4}$$

The overall of the controller is expressed as

$$u(t) = -\sum_{i=1}^{r} h_i(z(t))\boldsymbol{F}_i(x(t))\hat{x}(x(t)) \tag{5}$$

where $\boldsymbol{F}_i(x(t)) \in R^{m\times N}$ is polynomial matrices.

Combing the polynomial fuzzy model and controller, the total of the closed-loop control system can be described as

$$\begin{aligned} \dot{x}(t) = &\sum_{i=1}^{r}\sum_{j=1}^{r} h_i(z(t))h_j(z(t)) \\ &\times \{\boldsymbol{A}_i(x(t)) - \boldsymbol{B}_i(x(t))\boldsymbol{F}_j(x(t))\}\hat{x}(x(t)) \end{aligned} \tag{6}$$

B. Sum of Squares

A sum of squares polynomial $f(x(t))$ can be decomposed as $f(x(t)) = f_1^2(x(t)) + f_2^2(x(t)) + \cdots + f_m^2(x(t))$ where $f_1(x(t))$, $f_2(x(t))$, …, $f_m(x(t))$ are polynomial. Proposition 1 is utilized in Lemma 1 and Theorem 1.

Proposition 1 [16]: Let $\boldsymbol{G}(x(t)) \in R^{N\times N}$ be a symmetric polynomial matrix of degree $2d$, $\hat{x}(x(t))$ is a column vector, and $\boldsymbol{P}$ is a positive semidefinite matrix. In addition, the entries of $\hat{x}(x(t))$ are all monomial with degree no greater than d in $x(t) \in R^n$, and consider the following conditions.

1) *The polynomial* $\phi^T(t)\boldsymbol{G}(x(t))\phi(t)$ *equal to* $(\phi(t) \otimes \hat{x}(x(t)))^T \boldsymbol{P}(\phi(t) \otimes \hat{x}(x(t)))$ *where* $\otimes$ *denotes the Kronecker product.*
2) *The polynomial* $\phi^T(t)\boldsymbol{G}(x(t))\phi(t)$ *is an SOS, where* $\phi(t) \in R^N$.
3) $\boldsymbol{G}(x(t)) \geq 0$ *for all* $x(t) \in R^n$.

Then, $(1) \Leftrightarrow (2)$ and $(2) \Rightarrow (3)$.

C. Stability Conditions

In this section, x is instead of $x(t)$. $\boldsymbol{A}_i^k$ denotes the k^{th} row of $\boldsymbol{A}_i$. k_m is the row indices of $\boldsymbol{B}_i(x)$whose corresponding row is equal to zero, and define $\tilde{x} = (x_{k1}, x_{k2}, \cdots, x_{km})$ and $\boldsymbol{K} = \{k_1, k_2, \cdots, k_m\}$.

Lemma 1 [8]: If there exist a symmetric polynomial matrix $\boldsymbol{X}(\tilde{x}) \in R^{N\times N}$ and a polynomial matrix $\boldsymbol{M}_i(x) \in R^{m\times N}$ satisfy (7) and (8), the stability of (6) can be guaranteed.

$$\phi^T\left(\boldsymbol{X}(\tilde{x}) - \varepsilon_1(x)\boldsymbol{I}\right)\phi \text{ is SOS} \tag{7}$$

$$\begin{aligned} -\phi^T\Big(&\boldsymbol{Q}(x)\boldsymbol{A}_i(x)\boldsymbol{X}(\tilde{x}) - \boldsymbol{Q}(x)\boldsymbol{B}_i(x)\boldsymbol{M}_j(x) \\ &+ \boldsymbol{X}(\tilde{x})\boldsymbol{A}_i^T(x)\boldsymbol{Q}^T(x) - \boldsymbol{M}_j^T(x)\boldsymbol{B}_i^T(x)\boldsymbol{Q}^T(x) \\ &+ \boldsymbol{Q}(x)\boldsymbol{A}_j(x)\boldsymbol{X}(\tilde{x}) - \boldsymbol{Q}(x)\boldsymbol{B}_j(x)\boldsymbol{M}_i(x) \\ &+ \boldsymbol{X}(\tilde{x})\boldsymbol{A}_j^T(x)\boldsymbol{Q}^T(x) - \boldsymbol{M}_i^T(x)\boldsymbol{B}_j^T(x)\boldsymbol{Q}^T(x) \\ &- \sum_{k\in\boldsymbol{K}} \frac{\partial\boldsymbol{X}}{\partial x_k}(\tilde{x})\boldsymbol{A}_i^k(x)\hat{x}(x) \\ &\quad - \sum_{k\in\boldsymbol{K}} \frac{\partial X}{\partial x_k}(\tilde{x})\boldsymbol{A}_j^k(x)\hat{x}(x) + \varepsilon_{2ij}(x)\boldsymbol{I}\Big)\phi \text{ is SOS}, i \leq j \end{aligned} \tag{8}$$

where $\varepsilon_1(x) > 0$ is a nonnegative polynomial for $x \neq 0$. $\varepsilon_{2ij}(x) \geq 0$ is a nonnegative polynomial for all x. $\phi \in R^N$ is a vector that is independent of x. $\boldsymbol{Q}(x) \in R^{N\times n}$ is a polynomial matrix whose (i, j)th entry is given by $Q^{ij}(x) = (\partial\hat{x}_i / \partial x_j)(x)$. The stabilizing feedback gain $\boldsymbol{F}_i(x)$ is defined as $\boldsymbol{F}_i(x) = \boldsymbol{M}_i(x)\boldsymbol{X}^{-1}(\tilde{x})$.

Assume the $\boldsymbol{F}_i(x)$ is known, lemma 1 can be rewritten as follows.

Theorem 1: Assume $\boldsymbol{A}_i(x)$, $\boldsymbol{B}_i(x)$, and $\boldsymbol{F}_j(x)$ are known, the stability of (6) is guaranteed if there exist a positive semidefinite polynomial matrix $\boldsymbol{P}(\tilde{x})$ such that (9) and (10) hold.

$$\phi^T\left(\boldsymbol{P}(\tilde{x}) - \varepsilon_1(x)\boldsymbol{I}\right)\phi \text{ is SOS} \tag{9}$$

$$\begin{aligned} -\phi^T\Big(&\boldsymbol{P}(\tilde{x})\boldsymbol{Q}(x)\left(\boldsymbol{A}_i(x) - \boldsymbol{B}_i(x)\boldsymbol{F}_j(x)\right) \\ &+ \left(\boldsymbol{A}_i(x) - \boldsymbol{B}_i(x)\boldsymbol{F}_j(x)\right)^T\boldsymbol{Q}^T(x)\boldsymbol{P}(\tilde{x}) \\ &+ \boldsymbol{P}(\tilde{x})\boldsymbol{Q}(x)\left(\boldsymbol{A}_j(x) - \boldsymbol{B}_j(x)\boldsymbol{F}_i(x)\right) \\ &+ \left(\boldsymbol{A}_j(x) - \boldsymbol{B}_j(x)\boldsymbol{F}_i(x)\right)^T\boldsymbol{Q}^T(x)\boldsymbol{P}(\tilde{x}) \\ &+ \sum_{k\in\boldsymbol{K}} \frac{\partial\boldsymbol{P}}{\partial x_k}(\tilde{x})\boldsymbol{A}_i^k(x)\hat{x}(x) \\ &\quad + \sum_{k\in\boldsymbol{K}} \frac{\partial\boldsymbol{P}}{\partial x_k}(\tilde{x})\boldsymbol{A}_j^k(x)\hat{x}(x) + \varepsilon_{2ij}(x)\boldsymbol{I}\Big)\phi \text{ is SOS}, i \leq j \end{aligned} \tag{10}$$

where $\varepsilon_1(x) > 0$ is a nonnegative polynomial for $x \neq 0$. $\varepsilon_{2ij}(x) \geq 0$ is a nonnegative polynomial for all x. $\phi \in R^N$ is a vector that is independent of x. $\boldsymbol{Q}(x) \in R^{N\times n}$ is a polynomial matrix whose (i, j)th entry is given by $Q^{ij}(x) = (\partial\hat{x}_i / \partial x_j)(x)$.

III. Polynomial Fuzzy Control of Pendubot System

A. Pendubot System

Fig. 1 shows the structure of the Pendubot system. Fig. 2 shows the scheme of the Pendubot model.

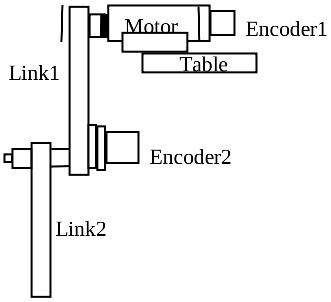


Fig. 1. The Pendubot system.

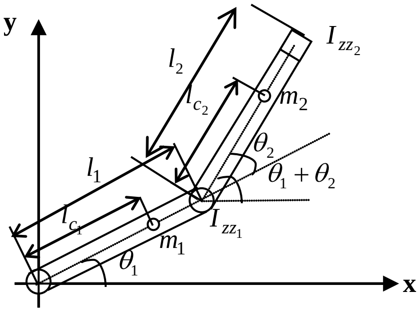


Fig. 2. The Pendubot scheme.

The motion equation of the Pendubot can be presented as

$$\boldsymbol{D}(\theta)\ddot{\theta}+\boldsymbol{C}(\theta,\dot{\theta})\dot{\theta}+\boldsymbol{G}(\theta)+\boldsymbol{F}(\dot{\theta})=\boldsymbol{\tau} \qquad (11)$$

where $\theta=[\theta_1 \quad \theta_2]^T$, $\boldsymbol{\tau}=[\tau_1 \quad 0]^T$ is the input vector,

$$\boldsymbol{D}(\theta)=\begin{bmatrix} D_{11} & D_{12} \\ D_{12} & D_{22} \end{bmatrix}, \boldsymbol{C}(\theta,\dot{\theta})=\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & 0 \end{bmatrix}, \boldsymbol{G}(\theta)=\begin{bmatrix} G_1 \\ G_2 \end{bmatrix},$$

$$\boldsymbol{F}(\dot{\theta})=[k_{\mu1}\dot{\theta}_1 \quad k_{\mu2}\dot{\theta}_2]^T$$

$$D_{11}=m_1 l_{c1}^2+m_2\{l_1^2+l_{c2}^2+2l_1 l_{c2}\cos(\theta_2)\}+I_{zz1}+I_{zz2}$$

$$D_{12}=m_2\{l_{c2}^2+l_1 l_{c2}\cos(\theta_2)\}+I_{zz2}$$

$$D_{22}=m_2 l_{c2}^2+I_{zz2} \qquad C_{11}=-2m_2 l_1 l_{c2}\dot{\theta}_2\sin(\theta_2)$$

$$C_{12}=-m_2 l_1 l_{c2}\dot{\theta}_2\sin(\theta_2) \qquad C_{21}=m_2 l_1 l_{c2}\dot{\theta}_1\sin(\theta_2)$$

$$G_1=-m_1 g l_{c1}\sin(\theta_1)-m_2 g\{l_1\sin(\theta_1)+l_{c2}\sin(\theta_1+\theta_2)\}$$

$$G_2=-m_2 g l_{c2}\sin(\theta_1+\theta_2)$$

TABLE I THE PARAMETER OF TWO-LINK ROBOT ARM.

Symbol	Value	Statement
m_1	$0.8293Kg$	The mass of link 1
m_2	$0.3402Kg$	The mass of link 2
l_1	$0.2032m$	The lengths of link1
l_2	$0.3841m$	The lengths of link 2
l_{c1}	$0.1551m$	The distances to the center of mass of link 1
l_{c2}	$0.1635m$	The distances to the center of mass of link 2
I_{zz1}	$0.005Kg\cdot m^2$	The moment of inertia of link 1
I_{zz2}	$0.0043Kg\cdot m^2$	The moment of inertia of link 2
$k_{\mu1}$	$0.00545Kg/s$	The friction constant of θ_1
$k_{\mu1}$	$0.00047Kg/s$	The friction constant of θ_2
g	$9.8ms^{-2}$	The gravity

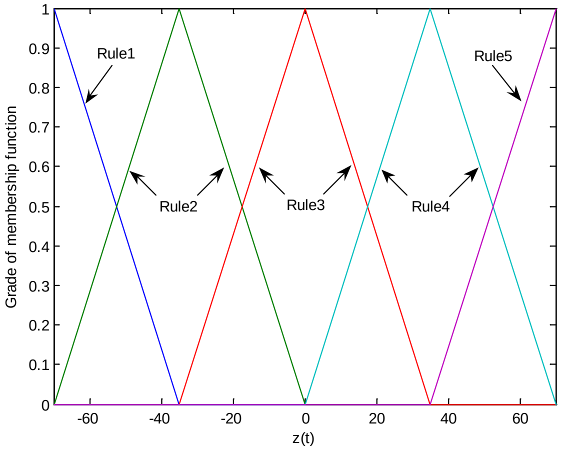


Fig. 3. The membership function of fuzzy model.

B. *Linearization and Fuzzy Modeling*

The motion equation of the Pendubot can be described as $\dot{x}(t)=f(x(t),u(t),t)$, where $x(t)=[\theta_1 \quad \dot{\theta}_1 \quad \theta_2 \quad \dot{\theta}_2]^T$ and $u(t)=\tau_1$.

Choose the premise variable $z(t)=x_3(t)$ with the following five-rule fuzzy model, and using Taylor series to linearize the system with the equilibrium point, we can obtain the matrices $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$ from different equilibrium point, where

$$\boldsymbol{A}_i:=\left.\frac{\partial f(x(t),u(t),t)}{\partial x}\right|_{x^i,u^i}, \boldsymbol{B}_i:=\left.\frac{\partial f(x(t),u(t),t)}{\partial u}\right|_{x^i,u^i}, \qquad (12)$$

x^i and u^i is the equilibrium point of rule i, and every equilibrium point must satisfy $\theta_1+\theta_2=90^\circ$.

Rule 1: If $z(t)$ is about -70°,
Then $\dot{x}(t)=\boldsymbol{A}_1 x(t)+\boldsymbol{B}_1 u(t)$

Rule 2: If $z(t)$ is about -35°,
Then $\dot{x}(t)=\boldsymbol{A}_2 x(t)+\boldsymbol{B}_2 u(t)$

Rule 3: If $z(t)$ is about 0,
Then $\dot{x}(t)=\boldsymbol{A}_3 x(t)+\boldsymbol{B}_3 u(t)$

Rule 4: If $z(t)$ is about 35°,
Then $\dot{x}(t)=\boldsymbol{A}_4 x(t)+\boldsymbol{B}_4 u(t)$

Rule 5: If $z(t)$ is about 70°,
Then $\dot{x}(t)=\boldsymbol{A}_5 x(t)+\boldsymbol{B}_5 u(t)$

The final output of the fuzzy model and controller are described by

$$\dot{x}(t)=\sum_{i=1}^{5} h_i(z(t))\{\boldsymbol{A}_i x(t)+\boldsymbol{B}_i u(t)\} \qquad (13)$$

and

$$u(t)=-\sum_{i=1}^{5} h_i(z(t))\boldsymbol{F}_i(x(t))x(t) \qquad (14)$$

where

$$\boldsymbol{A}_1=\begin{bmatrix}0 & 1 & 0 & 0\\ 13.3447 & -0.1439 & -4.1530 & 0.0160\\ 0 & 0 & 0 & 1\\ 23.5005 & 0.1854 & 46.0480 & 0.0557\end{bmatrix},$$

$$\boldsymbol{A}_2=\begin{bmatrix}0 & 1 & 0 & 0\\ 37.1419 & -0.6172 & -11.5592 & 0.0244\\ 0 & 0 & 0 & 1\\ -22.1188 & 0.2828 & 60.2461 & 0.0763\end{bmatrix},$$

$$\boldsymbol{A}_3=\begin{bmatrix}0 & 1 & 0 & 0\\ 50.1713 & -0.1850 & -15.6140 & 0.0294\\ 0 & 0 & 0 & 1\\ -51.8109 & 0.3411 & 69.4861 & 0.0893\end{bmatrix},$$

$$\boldsymbol{B}_1=\begin{bmatrix}0\\ 26.3785\\ 0\\ -34.0173\end{bmatrix},\quad \boldsymbol{B}_2=\begin{bmatrix}0\\ 30.6778\\ 0\\ -51.8831\end{bmatrix},\quad \boldsymbol{B}_3=\begin{bmatrix}0\\ 33.9453\\ 0\\ -62.5894\end{bmatrix},$$

$$\boldsymbol{A}_4=\boldsymbol{A}_2,\ \boldsymbol{B}_4=\boldsymbol{B}_2,\ \boldsymbol{A}_5=\boldsymbol{A}_1,\ \boldsymbol{B}_5=\boldsymbol{B}_1$$

IV. Results

A. Regulation Control

The following gains are obtained by LMI-based T-S fuzzy control.

$$\boldsymbol{G}_1=\begin{bmatrix}-1587.9 & -303.2 & -1507.3 & -237.1\end{bmatrix}$$
$$\boldsymbol{G}_2=\begin{bmatrix}-1096.6 & -209.5 & -1042.0 & -163.8\end{bmatrix}$$
$$\boldsymbol{G}_3=\begin{bmatrix}-437.9 & -83.8 & -417.4 & -65.5\end{bmatrix}$$
$$\boldsymbol{G}_4=\boldsymbol{G}_2,\ \boldsymbol{G}_5=\boldsymbol{G}_1$$

where the eigenvalues of $\boldsymbol{P}$ are 50691.340, 0.044, 0.969, 0.667, respectively.

By lemma 1, the feedback gains can be obtained as follows.

$$\begin{aligned}F_1(x)=&F_{1a}+F_{1b}x_1+F_{1c}x_2+F_{1d}x_3+F_{1e}x_4+F_{1f}x_1^2+F_{1g}x_1x_2\\&+F_{1h}x_1x_3+F_{1i}x_1x_4+F_{1j}x_2^2+F_{1k}x_2x_3+F_{1l}x_2x_4\\&+F_{1m}x_3^2+F_{1n}x_3x_4+F_{1o}x_4^2\end{aligned}$$

$$\begin{aligned}F_2(x)=&F_{2a}+F_{2b}x_1+F_{2c}x_2+F_{2d}x_3+F_{2e}x_4+F_{2f}x_1^2+F_{2g}x_1x_2\\&+F_{2h}x_1x_3+F_{2i}x_1x_4+F_{2j}x_2^2+F_{2k}x_2x_3+F_{2l}x_2x_4\\&+F_{2m}x_3^2+F_{2n}x_3x_4+F_{2o}x_4^2\end{aligned}$$

$$\begin{aligned}F_3(x)=&F_{3a}+F_{3b}x_1+F_{3c}x_2+F_{3d}x_3+F_{3e}x_4+F_{3f}x_1^2+F_{3g}x_1x_2\\&+F_{3h}x_1x_3+F_{3i}x_1x_4+F_{3j}x_2^2+F_{3k}x_2x_3+F_{3l}x_2x_4\\&+F_{3m}x_3^2+F_{3n}x_3x_4+F_{3o}x_4^2\end{aligned}$$

$$\begin{aligned}F_4(x)=&F_{4a}+F_{4b}x_1+F_{4c}x_2+F_{4d}x_3+F_{4e}x_4+F_{4f}x_1^2+F_{4g}x_1x_2\\&+F_{4h}x_1x_3+F_{4i}x_1x_4+F_{4j}x_2^2+F_{4k}x_2x_3+F_{4l}x_2x_4\\&+F_{4m}x_3^2+F_{4n}x_3x_4+F_{4o}x_4^2\end{aligned}$$

$$\begin{aligned}F_5(x)=&F_{5a}+F_{5b}x_1+F_{5c}x_2+F_{5d}x_3+F_{5e}x_4+F_{5f}x_1^2+F_{5g}x_1x_2\\&+F_{5h}x_1x_3+F_{5i}x_1x_4+F_{5j}x_2^2+F_{5k}x_2x_3+F_{5l}x_2x_4\\&+F_{5m}x_3^2+F_{5n}x_3x_4+F_{5o}x_4^2\end{aligned}$$

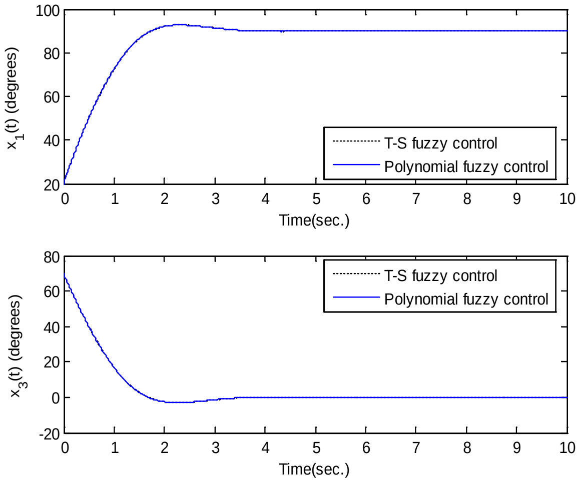


Fig. 4. The system responses of regulation control.

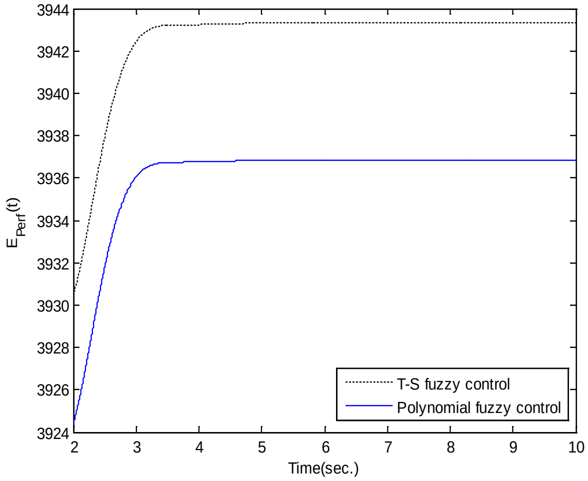


Fig. 5. Comparison of the system performance.

where the eigenvalues of $\boldsymbol{P}$ are 24297.160, 0.004, 0.090, 0.067, respectively. The feedback gains $F_{ia}, F_{ib}, \cdots, F_{io}$ can be obtained by using the SOSTOOL. System responses are shown in Fig. 4 with $x(0)=[20^\circ \quad 11.46 \quad 70^\circ \quad 11.46]^T$. The system performance is measured by

$$E_{Perf.}(t)=\int \left(e^T(t)e(t)\right)dt\,, \qquad (15)$$

where $e(t)=EP-[x_1 \quad x_3]$. $EP=[90^\circ \quad 0^\circ]$ is a equilibrium point. Fig. 5 shows the variation of system performance.

B. Step Tracking

Let the closed-loop system track a reference signal, the system responses are shown in Fig. 6.

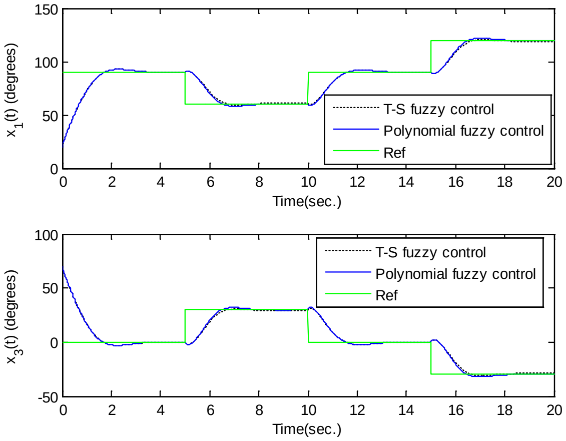


Fig. 6. The state variations of step tracking control.

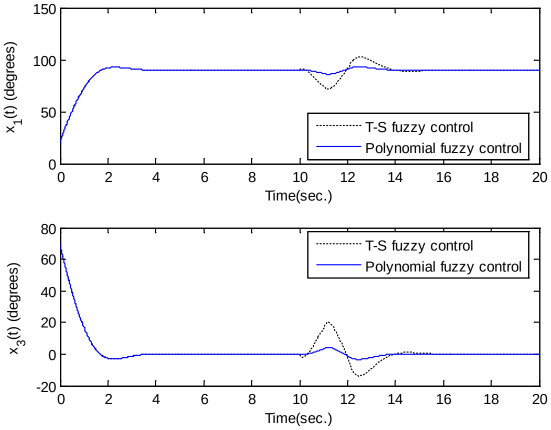


Fig. 7. The state responses with disturbance.

C. Disturbance Rejection

Fig. 7 shows the system response with disturbance where the disturbance is given as follows

$$v(t) = \begin{cases} 10, & 10 \le t < 11 \\ -10, & 11 \le t < 12 \\ 0, & \text{other} \end{cases} \tag{16}$$

D. Feasible Area with Uncertainty

Let the fuzzy system includes model uncertainty, it can be rewritten as

$$\dot{x}(t) = \sum_{i}^{r} h_i(z(t)) \{ \boldsymbol{A}_i x(t) - (\boldsymbol{B}_i + \Delta \boldsymbol{B}_i) u(t) \} \tag{17}$$

where $\Delta \boldsymbol{B}_i = [0 \quad \Delta_1 \times \boldsymbol{B}_i(2,1) \quad 0 \quad \Delta_2 \times \boldsymbol{B}_i(4,1)]^T$, Δ_1 and Δ_2 are the uncertainty percentage. $\boldsymbol{B}_i(2,1)$ deotes the (2,1)th entry of $\boldsymbol{B}_i$. $\boldsymbol{B}_i(4,1)$ denotes the (4,1)th entry of $\boldsymbol{B}_i$.

Fig. 8 shows the feasible area of $\boldsymbol{P} \ge 0$ between SOS-based polynomial fuzzy control and LMI-based T-S fuzzy control.

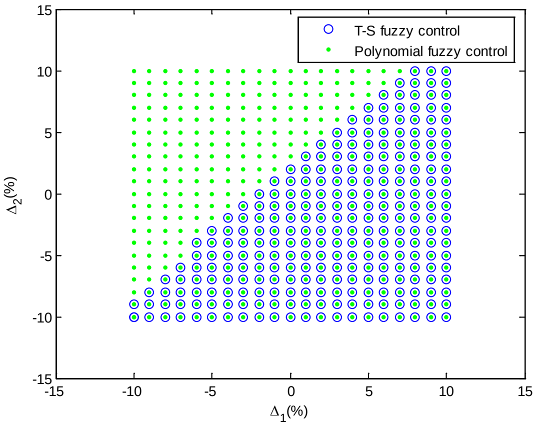


Fig. 8. Feasible area by different fuzzy control approach.

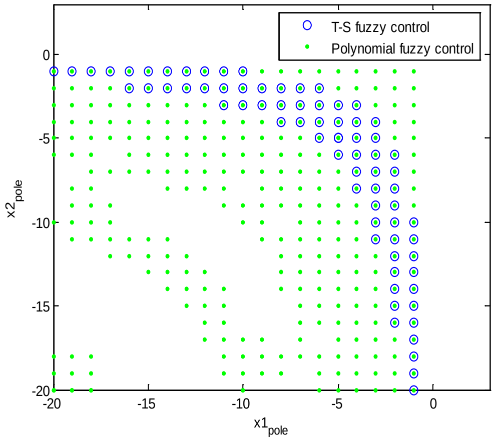


Fig. 9. Feasible area of system performance.

E. Feasible Area of System Performance

From section IV-A, the stability of the Pendubot system can be guaranteed by lemma 1, but there is no visible difference in system responses between SOS-based polynomial fuzzy control and LMI-based T-S fuzzy control. In this section, consider the feedback gains are determined by the pole placement method first such that the system performance requirement can be achieved. Then, the stability of the Pendubot system can be ensured by Theorem 1. Fig. 9 shows the feasible area of $\boldsymbol{P} \ge 0$ between Theorem1 and T-S fuzzy control.

V. Conclusion

The controller design of the Pendubot by using polynomial fuzzy model and sum-of-squares approach was proposed. The stability of the Pendubot can be guaranteed by SOS-based polynomial fuzzy control. According to Fig. 7, the control gains obtained from polynomial fuzzy control is more effective in disturbance rejection. Fig.8 illustrates that the SOS-based polynomial fuzzy control is a better approach to deal with the fuzzy uncertain model. Last, the performance requirement can be achieved by the pole placement method, and Fig. 9. shows the polynomial fuzzy control with Theorem 1 is more useful to determine the feasible area for the stability of the Pendubot system than T-S fuzzy control.

References

[1] K. Tanaka and H. O. Wang, *Fuzzy Control Systems Design and Analysis: A Linear Matrix Inequality Approach*. New York: Wiley, 2008.

[2] H. K. Lam and S. C. Tan, "Stability analysis of fuzzy-model-based control systems: application on regulation of switching DC–DC converter," *IET Journal of Control Theory & Applications*, vol. 3, no. 8, pp. 1093-1106, 2009.

[3] M. Biglarbegian, W. W. Melek and J. M. Mendel, "On the stability of interval type-2 TSK fuzzy logic control systems," *IEEE Trans. Syst., Man, Cybern.*, vol. 40, no. 5, pp. 798 - 818, June 2010.

[4] M. Narimani, H. K. Lam, R. Dilmaghani, and C. Wolfe, "LMI-based stability analysis of fuzzy-model-based control systems using approximated polynomial membership functions," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 3, pp. 713–724, Jun. 2011.

[5] Y. J. Chen, H. Ohtake, K. Tanaka, W.-J. Wang and H. O. Wang, "Relaxed stabilization criterion for T-S fuzzy systems by minimum-type piecewise Lyapunov-function-based switching fuzzy controller," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1166-1173, Dec. 2012.

[6] Qiu, G. Feng, and H. Gao, "Static-output-feedback H∞ control of continuous time T-S fuzzy affine systems via piecewise Lyapunov functions," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 2, pp. 245–261, Apr. 2013.

[7] J.-Y. Chang and C.-T. Lin, "A TSK-type-based self-evolving compensatory interval type-2 fuzzy neural network (TSCIT2FNN) and its applications," *IEEE Trans. Industrial Electronics*, vol. 61, pp. 447 - 459, Jan. 2014.

[8] K. Tanaka, H. Yoshida, H. Ohtake, and H. O. Wang, "A Sum-of-Squares Approach to Modeling and Control of Nonlinear Dynamical Systems with Polynomial Fuzzy System," *IEEE Trans. Fuzzy Systems*, vol. 17, No. 4, Aug. 2009.

[9] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo, *SOSTOOLS: Sum of Squares Optimization Toolbox for MATLAB*, Version 2.00, California Inst. Technol., Pasadena, 2004.

[10] H. K. Lam, "Stabilization of nonlinear systems using sampled-data output-feedback fuzzy controller based on polynomial-fuzzy-model-based control approach," *IEEE Trans. Syst., Man, Cybern.*, vol. 42, no. 1 pp. 258 - 267, Feb. 2012.

[11] K. Tanaka, H. Ohtake, T. Seo, M. Tanaka, and H. O. Wang, "Polynomial fuzzy observer designs a sum-of-squares approach," *IEEE Trans. Syst., Man, Cybern.*, vol. 42, no. 5, pp. 1330 - 1342, Oct. 2012.

[12] H. K. Lam, C. Liu, L. Wu and X. Zhao, "Polynomial fuzzy-model-based control systems: stability analysis via approximated membership functions considering sector nonlinearity of control input," *IEEE Trans. Fuzzy Systems*, vol. 23, no. 6, pp. 2202-2214, 2015.

[13] K. Tanaka, M. Tanaka, Y. J. Chen, and H. O. Wang, "A new Sum-of-Squares design framework for robust control of polynomial fuzzy systems with uncertainties," *IEEE Trans. Fuzzy Syst.*, vol. 1, no. 11, pp. 1, April. 2015.

[14] G. R. Yu, Y. C. Huang and C. Y. Cheng, "Robust H∞ controller design for polynomial fuzzy control systems by sum-of-squares approach," *IET Control Theory Appl.*, vol. 10, no.11, pp. 1684 - 1695, Sep. 2016.

[15] X. Li, H. K. Lam, F. Liu and X. Zhao, "Stability and stabilization analysis of positive polynomial fuzzy systems with time delay considering piecewise membership functions," *IEEE Trans. Fuzzy Systems*, vol. 25, no. 4, pp. 958-971, 2017.

[16] S.Prajna, A. Papachristodoulou, and F. Wu, "Nonlinear Control Synthesis by Sum of Squares Optimization: A Lyapunov-Based Approach," *in Proc. Asian Control Conf.*, pp. 157-165, Feb. 2004.

Gwo-Ruey Yu received the Ph.D. degree in Electrical Engineering from the University of Southern California, Los Angeles, in 1997. He is currently a Professor of Electrical Engineering Department and the Director of Elegant Power Application Research Center, National Chung Cheng University, Taiwan. Dr. Yu is respectively the recipients of the Best Paper Award of IEEE 2017 International Automatic Control Conference, the Advisor Award of Robotic Society of Taiwan in 2018, the Best Paper Award in Application of IEEE 2016 International Conference on Fuzzy Theory and It's Applications, the Outstanding Paper Award of IEEE 2016 International Automatic Control Conference, the Best Paper Award of 2018 International Conference on Advanced Robotics and Intelligent Systems. His research interests include intelligent robots, automatic control based on artificial intelligence, and renewable energy systems.

Yu-Shan Chiu received the B.S. degree from Chung Yuan Christian University, Taoyuan City, Taiwan, in 2017. She is currently a student at Graduate Institute of Opto-Mechatronics, National Chung Cheng University, Chia-Yi, Taiwan. She received the Second Place Prize of the Best Paper Award of 2018 International Conference on Advanced Robotics and Intelligent Systems, in 2018. Her research interests include intelligent robots, fuzzy systems, and intelligent control.

Yu-Chia Huang received the B.S. degree in electrical engineering from I-Shou University, Taiwan, in 2007, the M.S. degree in electrical engineering from National Ilan University, Taiwan, in 2009, and the Ph.D degree in electrical engineering from National Taiwan Ocean University, Taiwan, in 2017. He received the Outstanding Paper Award of IEEE 2016 International Automatic Control Conference, in 2016. His research interests include intelligent systems and control, and nonlinear system control.

Meaningful Data Augmentation under Unbalanced Data in End-to-end Learning for Autonomous Driving

Hong-Ming Chen and Shun-Feng Su, *Fellow, IEEE*

Abstract- **In this study, learning methodologies for end-to-end autonomous driving are considered. Previous work has demonstrated the possibility of modern deep neural networks (DNNs) to mimic human driving. This study is aimed at exploring the use of a monocular camera for driving in complex and unstructured paths such as messy hallways or off-road trails. Existing driving DNN architecture is employed with different data manipulation strategies that consists of dataset balancing, data preprocessing and data augmentation. To the best of evaluation, a series of experiments are conducted on the Udacity driving simulator for quantifying the course following accuracy. Usually, there are more straight line then taking tacking the turn. This creates unbalance issues in the driving dataset obtained. In our study, meaningful data augmentation is considered to create more driving data by simulating left and right driving image. With appropriately assigned steering commands, as shown in our experiments, augmented data help improvement of the learning model to be more robust in autonomous driving. In the study, a new objective function is also considered in learning so as to increase sharp turn stability in driving. With this new objective function, the model used can learn under-represented sharp-turn scenario. This makes the system able to drive on racing tracks and even narrow mountain roads. Finally, by using the VisualBackProp technique, the road features learned from training data are able to be determined and to evaluate end-to-end learning systems.**

Index Terms—End-to-end learning, Autonomous driving, Data augmentation, Robotics platform.

I. Introduction

NOWDAYS, the majority of autonomous vehicles have been developed to be able to successfully navigate through an urban environment. However, most of the autonomous systems consist of overly complex and expensive technical components such as object perception, localization, trajectory planner, and control system. These systems require costly sensors, high definition map, and powerful processors to be able to operate. Until nearly a decade, there have been breakthroughs on hardware technologies. Since then, neural network-driven learning has made a major breakthrough in visual recognition [1-3]. Neural network has regained everyone's attention. Deep Convolutional Neural Networks can solve many pattern recognition and computer vision problems and set all sorts of new records in various competitions. Deep learning is able to learn the best representations from raw input data without any handcrafted features [4,5]. Among them, Convolutional Neural Networks (CNNs) are the most powerful deep learning model, achieving several state-of-the-art results in ImageNet challenge using AlexNet [1]. This success was employed in autonomous driving in lane mark and vehicle detection in [2].

End-to-end deep learning for autonomous vehicles is to learn driving actions directly from image data obtained with the use of deep learning schemes. Such an approach removes all those stages in traditional autonomous vehicles and replaces them with just a single neural network. Without hand-engineering, deep learning schemes self-optimize the system based on data. However, this data-driven network requires a huge amount of data that cover all driving scenarios to learn driving policies from a human driver. Unlike other classification problems that require human labeling, fortunately, collecting driving data and corresponding control commands for end-to-end autonomous driving is not a heavy task. Human driving data can be directly recorded on the road. This property makes end-to-end driving feasible and valuable for both academic and industrial research.

In this paper, the study on training strategies of end-to-end autonomous driving systems is reported. The proposed training strategies consist of meaningful data augmentation and new objective function for unbalanced data. A new loss function is proposed in this study to resolve sharp turn failure. From our experiments, it is evident that the proposed loss function indeed can result in successfully driving in various situations. A series of experiments are conducted to optimize hyper-parameters of the proposed training strategy with the use of the Udacity Driving Simulator [6]. Furthermore, to effectively evaluate the learning performance, the proposed training strategies uses VisualBackProp[22] techniques to visualize what the neural network has learned from training data. This paper presents our study on the end-to-end learning approach for directly generating proper steering angles from images obtained from a front-face camera. The driving model is trained and evaluated by a self-recording dataset in the simulator. The following contributions of this work are concluded:

•**A training methodology for an end-to-end autonomous driving system is considered and verified.** The methodology of how to train an end-to-end driving system is described in details including data preparation strategy, network architecture and hyper-parameter.

•**Meaningful data augmentation techniques for robust driving that emulates different positions of the car on the lane.** Unlike recording three different orientation from cameras as NVIDIA paper proposed, our system uses only one-center front-face camera. The perspective transformation is

Hong-Ming Chen and Shun-Feng Su are with the Department of Electrical Engineering, National Taiwan University of Science and Technology, 43 Keelung Road, Section 4, Taipei, Taiwan

(Corresponding Author: Shun-Feng Su, email: sfsu@mail.ntust.edu.tw)

considered on center driving images to create left and right driving images. In our study, the unbalanced data problem is also tackled.

•**New objective function that increases robustness as car driving through sharp turns.** In practice, more data in straight than that of taking sharp turns on the road. Due to this fact, the driving data collected are unbalanced. Common objective functions (mean square error (MSE) and mean absolute error (MAE)) are to match the majority of dataset, namely the straight driving data. As a result, the network learned has more errors on a sharp turn scenario. A new objective function is proposed to punish more on sharp turns. The new objective function indeed can resolve the above problem.

The structure of the remainder of this paper is as follows: Section 2 introduces related work of traditional approaches and deep learning approaches for autonomous vehicles. Section 3 describes the details of the proposed training strategies and the way of implementation, including data augmentation and the objective function proposed. Section 4 discusses the experimental results and in Section 5, the conclusions of this work are given.

II. Related Work

Nowadays, the development of autonomous driving can be divided into two distinct methods, the traditional robotics approach and the deep learning approach. Over the years, various types of perception systems and their optimization techniques have been proposed for autonomous navigation. The robotics approach decomposes the driving into individual tasks, such as lane detection [1], localization and motion planning. Recent work on autonomous driving is about using deep neural networks for image-based localization [9, 10] and trajectory planning [11]. Those techniques are briefly introduced in the following.

In order to achieve autonomous driving, researchers in the field of robotics will first derive a series of motion models for vehicles to achieve robust vehicle control.

Among the common control methods, MPC has achieved success in semi-autonomous driving and autonomous driving [12-14]. The MPC scheme proposed uses a dynamic vehicle model [15] combined with linear [14] and tire models [12]. To have autonomous driving, the surrounding environment of the self-driving car needs to be properly identified. The moving objects are recognized by using multiple radars, LIDAR and visual sensors mounted on the vehicle. Based on the distance between the moving object and the shape of the object sensed, the vehicle is able to identify other vehicles, pedestrians and its surrounding scene. In [16], the CNN network is employed to identify signs on the road. Support vector machine (SVM) is considered to identify pedestrian movements in [17]. In DeepLanes [18], a camera is used to estimate the lane position on both sides of the car. It can be seen that those are to achieve certain tasks in autonomous driving.

On the contrast, a deep learning approach is to replace all these processes by a single deep neural network. The system learns how to drive by mimicking human behavior. End-to-end learning for self-driving can date back to 1989. Pomerleau [8] built a control system for a vehicle with a Neural Network system called ALVINN. It applied an end-to-end fully-connected neural network that successful drive a car on public roads. LeCun proposed DAVE [19] in 2005. DAVE used a convolution neural network for end-to-end obstacle avoidance. In 2016, Bojarski et al. [7] demonstrated a DAVE-2 system can successful drive a full sized vehicle. It used modern deep learning techniques and a trained convolution neural network to infer appropriate steering angles given as input only front-face camera images. The above work has proven the prospect of a deep end-to-end architecture that is directly perceived in autonomous driving. Most notably, in [7], by recording driving video and vehicle steering angles, standard convolution neural networks are used only to successfully drive full-sized vehicles on typical roads, even in bad weather conditions. These end-to-end systems have demonstrated the feasibility of highly autonomous end-to-end imitation learning.

Figure 1. Driving image from **Campus hallway** (left) and **Udacity driving simulator** (right).

Figure 2. Udacity Driving Simulator.

Our work is partly inspired by, and is most closely related to, DARPA Autonomous Vehicle research [19] in which a subscale radio control (RC) car drove through a narrow alley. Recent work [20] proposed 1/5 scale AutoRally truck platform that carry a full desktop computer and perform autonomous racing on real world track. Today's hardware are much powerful to explore deeper neural networks on embedded devices. [3] demonstrated such hardware systems are able to run end-to-end network and drive their 1/5 scale truck in the real world. Above successful research motivated us to build a small-scale vehicle to explore end-to-end learning in pursuit of autonomous driving.

III. Meaningful Data Augmentation for Self-Driving Car

This section describes the proposed data augmentation method for image captured on driving by a center car camera so as to resolve unbalance dataset problems. The method of estimating new steering angles on augmented driving images is also described. Furthermore, a novel objective function for robustly driving is also proposed in this section.

A. System Overview

Our study demonstrates the end-to-end learning on driving in a computer simulator and real-world driving. Both experiments

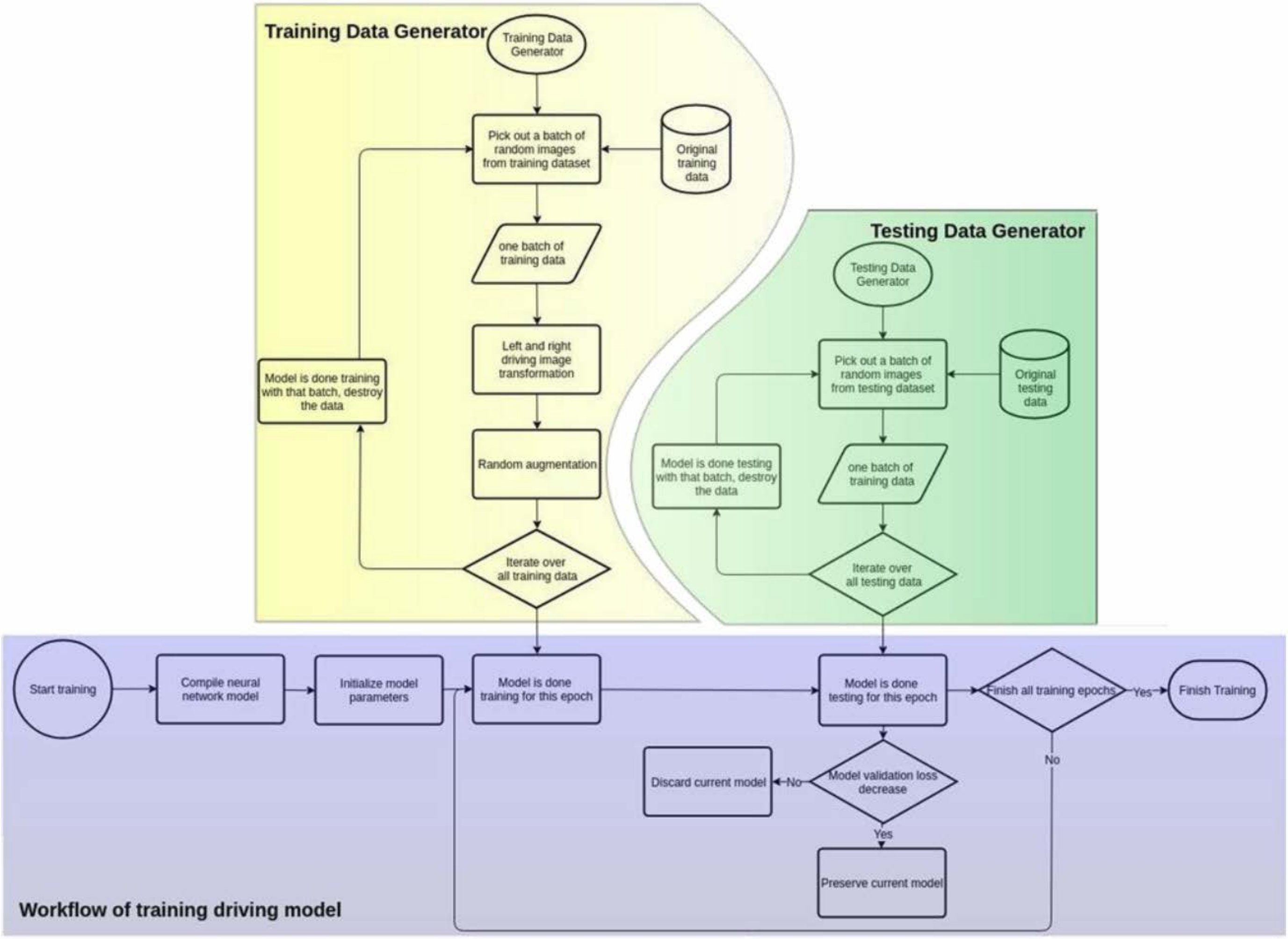

Training Data Generator
Training Data Generator
Pick out a batch of random images from training dataset
Original training data
one batch of training data
Left and right driving image transformation
Random augmentation
Iterate over all training data
Model is done training with that batch, destroy the data
Testing Data Generator
Testing Data Generator
Pick out a batch of random images from testing dataset
Original testing data
one batch of training data
Model is done testing with that batch, destroy the data
Iterate over all testing data
Start training
Compile neural network model
Initialize model parameters
Model is done training for this epoch
Model is done testing for this epoch
Finish all training epochs
Yes
Finish Training
No
Model validation loss decrease
No
Discard current model
Yes
Preserve current model
Workflow of training driving model

Figure 3. Flowchart of training pipeline.

are in unstructured environments, as shown in Figure 1. An end-to-end deep neural network is employed for predicting steer control commands from camera images. In data collection, a third person environment view (center camera) with a resolution of 800x600 pixels is presented to the human driver. The driver uses the keyboard to control the simulated vehicle throttle and moves the mouse slightly adjust steering wheel, as shown in Figure 2.

In the learning phase, the car learns to end-to-end control by CNN based on driving image and steering angle pairs in the training dataset. Our training and testing system as shown in Figure 3. In the figure, the purple part is the main program about training the neural network. The yellow and green parts are the training and the testing data generators. Those three programs are executed at the same time for real-time generation of augmented training data.

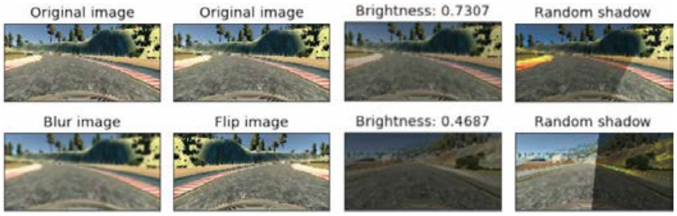


Figure 4. Typical Data augmentation.

B. *Data Augmentation*

As mentioned, in the training of our study, data argument is implemented to enrich the data distribution. The data augmentation pipeline used in our study consists of typical augmentation methods and some proposed data transformation. Those schemes used are random horizontal flips with appropriate steer command changes (20%), random Gaussian blurriness (20%), random brightness (20%), random shadow (20%) and proposed random image viewpoint transformations with appropriate steer command changes (20%). An example of the former 4 schemes are shown in Figure 4.

If training dataset only contains the central camera images, the car may soon leave the track and be crashed. This is because dataset only record ideal driving scenario and the neural network does not know what to do when vehicle is slightly drifting. One way to resolve this problem is to record recover

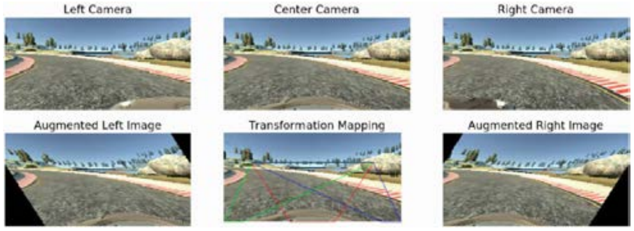


Figure 5. Source and destination points of left/right perspective projection.

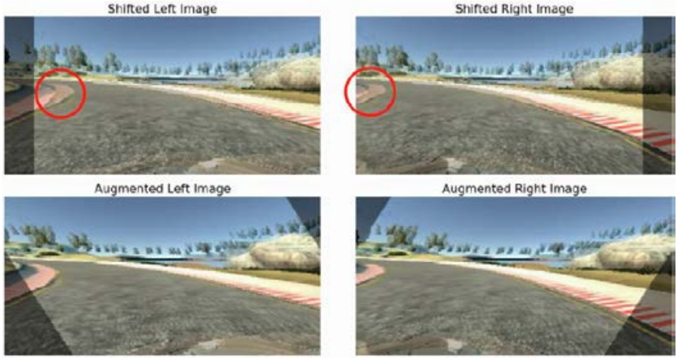


Figure 6. Overlap perspective transformed image with left/right camera image.

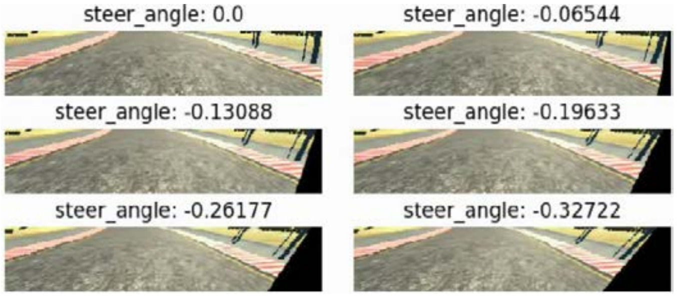


Figure 7. Perspective transformation mapping.

driving while driving from the roadside to the middle of the road. Another way is to use three cameras to simultaneously record center and both sides of the road image. Inspired by the work in [7] where three front-facing cameras are mounted on the car and training data collected from the three different perspectives on a road. By assigning appropriate steering commands to the left and right driving images, the situations of the vehicle drifting on the road can be emulated and the car driving commands assigned are to steer the car back to the center of the lane.

The simplest way to create augmented left and right images is horizontally shifted entire image for 30 pixels. However; this method will introduce distortion on road features. As a result, perspective transform is applied for the center driving images to emulate the car on different positions on the road and the corresponding steering commands are assigned to those transformed images. Figure 5 illustrates transformation mapping from center to left/right images, in which the red quadrangle stands for source points. The source points are project to destination points (green/blue points) for left/right driving images. The proposed method is compared with directly shift in Figure 6. By overlapping images real left/right camera image and simulated left/right images, the proposed method has the same road feature without distortion. An image generator is implemented that takes the original driving images as input and generates multiple perspective driving images. By using the simulated left and right driving images, the dataset size became at least 3 times larger. In our implementation, randomly transformed driving images is also considered for generate more driving images. The random transformation is from 0 to 5 (degree) offset of original images as shown in Figure 7.

TABLE I. METHODS OF FILLING IN BACKGROUND AREA.

Methods	Sample Images
3 cameras training (left/center/right)	
Original black color center camera with aug	
Random color center camera with aug	
Random pixel color center camera with aug	
Replicate border color center camera with aug	

TABLE II. METHODS OF FILLING IN BACKGROUND AREA.

Metrics	Autonomy (3/3 laps)		RMSE		CTE	
Track	Lake	Jungle	Lake	Jungle	Lake	Jungle
3 cameras training (left/center/right)	3/3	3/3	0.17964	0.57641	11509.30	36137.97
Original black color center camera with aug	2.047/3	3/3	0.12009	0.54753	8880.43	36561.22
Random color center camera with aug	3/3	3/3	0.12606	0.55192	9067.61	**34718.61**
Random pixel color center camera with aug	3/3	3/3	**0.12251**	**0.54640**	9109.62	36094.34
Replicate border color center camera with aug	3/3	3/3	0.13486	0.56370	**8575.66**	36235.76

TABLE III. RECOVER STEERING ANGLE EXPERIMENTS.

Metrics	Autonomy (3/3 laps)		RMSE		CTE	
Track	Lake	Jungle	Lake	Jungle	Lake	Jungle
Shift 1.25 deg.	0.165/3	0.365/3	0.1224	**0.5318**	-	-
Shift 2.5 deg.	3/3	0.285/3	0.1222	0.5352	10634.6787	-
Shift 5 deg.	**3/3**	**3/3**	0.1266	0.5423	10264.1566	**37577.7391**
Shift 10 deg.	3/3	1.65/3	0.1269	0.5451	10491.9161	-
Shift 15 deg.	3/3	0.66/3	0.1224	0.5467	11186.1015	-

After perspective transformation, in the new images there will be some black background areas on the corners. Black area stands for zero pixel value in RGB color model. Keeping the values as zeros will introduce some bias to the network learning. In order to check whether the neural network remembers the black areas of the augmented images, a series of experiments is conducted to confirm effectiveness on those black background areas. The following methods for filling the background area are compared. The first one is to fill random color on the background area. The second approach is to fill random pixel color on the background area. Finally, to replicate border color values on the background area. The results are shown in Table 1. Our baseline method is filling black color on background area. The results of the proposed approach is also compared with three cameras setup and are tabulated In Table 2. The result shows that the proposed data augmentation can indeed improve driving neural network performance.

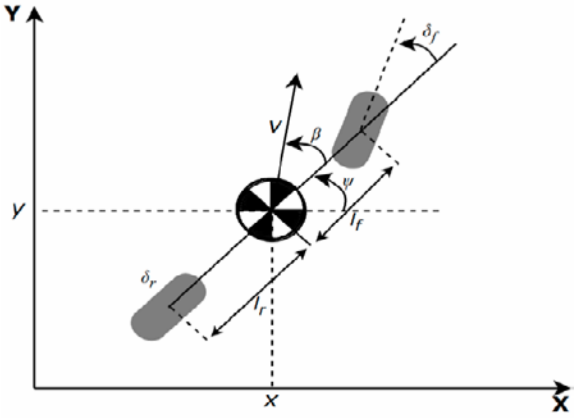


Figure 8. Kinematic Bicycle Model.

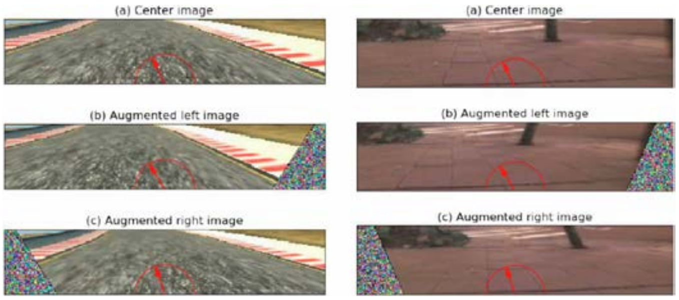

(a) Center image
(a) Center image
(b) Augmented left image
(b) Augmented left image
(c) Augmented right image
(c) Augmented right image

Figure 9. Visualize shift steer angle on augmented images.

C. *Estimate Steering Angle*

In our preliminary experiment, the recover steering angle was treated as a hyperparameter. As shown in Table 3, several experiments have been conducted to determine the best recover steering angle.

However, to assign a fixed recover steering angle for all the augmented driving images is not reasonable. The steering angle should depend on the current vehicle speed. As a result, the bicycle kinematic model is adopted to formulate recover steering angles. First, the vehicle model can simplify to a simple bicycle model as Figure 8. An example is shown in Figure. 9. When the camera view point was shifted leftwards (namely, the car was on the left side of the track), the steering command is corrected to steer back to the center.

During training, appropriate steering commands are calculated for augmented left and right camera images by kinematic bicycle model [21, 15]. As shown in Figure 10, by simplifying vehicle kinematic model to a bicycle kinematic model, each pair of wheels is represented as a vehicle kinematic system. Those three cameras are on the center and left and right, respectively, on the car model. The blue and the green vehicles represents the car drifting on the road for left and right views. The front wheel of the blue and the green vehicles is adjusted for turning back to the center line.

The equations of motion that describe the kinematic bicycle model are:

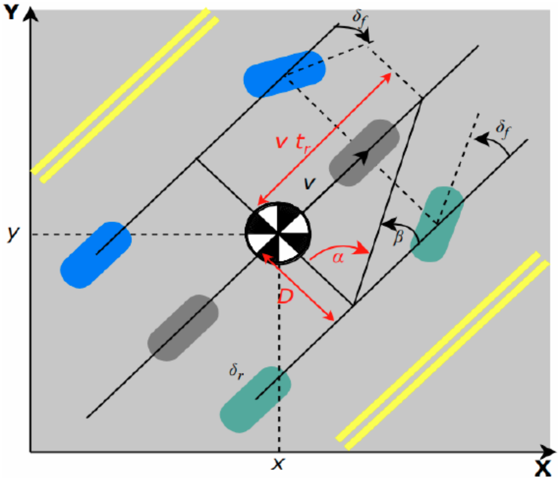


Figure 10. Kinematic model of 3 cameras setup.

$$\dot{x} = v\cos(\psi+\beta) \qquad \dot{y} = v\sin(\psi+\beta)$$
$$\dot{\psi} = \frac{v}{l_r}\sin(\beta) \qquad \beta = \tan^{-1}\left(\frac{l_r}{l_f+l_r}\tan(\delta_f)\right) \tag{1}$$

where (x, y) is the coordinate of the center of the mess in the global coordinate (X, Y), *v* is the speed of the vehicle, ψ is the orientation of the vehicle in the global coordinate, β is the steering angle in the vehicle coordinate, and δ_f is the exact steering angle of the front wheel. l_f and l_r are the distance from the center of the mess to the rear/front wheel, respectively. Since the augmented camera are shifted leftward/ rightward from the center, the distance *D* between the center camera and the augmented camera position can be inferred. Given the current vehicle speed *v* and defining the elapsed time of recovery t_r, the steer angle δ_f can be determined by solving the following equation

$$\alpha = \cos^{-1}\left(D\Big/\sqrt{D^2+(vt_r)^2}\right)$$
$$\beta = {}^{\pi}\!\big/_{2} - \alpha \tag{2}$$
$$\delta_f = \tan^{-1}\left(\frac{l_r+l_f}{l_r}\tan(\beta)\right)$$

These recover commands will steer the vehicle toward the middle of the lane by emulating the vehicle driving from different positions on the road. The DNN model is trained to associate the visual inputs with the correct steering commands.

D. *Objective Function*

After training the driving network, the trained network can complete three cycles of autonomous driving on the lake track. However, in the jungle track, because the jungle track consists with downhill, sharp turns nearly 160 degrees and narrow road, the trained driving network can no longer complete three autonomous driving at a high speed. The reason is mainly that the training dataset has been balanced, but the result of the MSE training will make the minority of sharp turn scenarios to have a relatively large error compared with the majority of straight driving. When the simulated vehicle driving in a sharp turn, it is easy to hit the guardrail due to insufficient steering.

TABLE IV. LOSS FUNCTION EXPERIMENTS.

Metrics	**Autonomy (3/3 laps)**		**RMSE**		**CTE**	
Track	**Lake**	**Jungle**	**Lake**	**Jungle**	**Lake**	**Jungle**
MSE	3/3	1.5/3	**0.15157**	**0.52813**	10292.2105	-
MAE	3/3	1/3	0.15365	0.53233	10250.3850	-
MAPE	0/3	0/3	-	-	-	-
MLSE	0/3	0/3	-	-	-	-
Proposed loss	3/3	3/3	0.16458	0.59250	**9611.4355**	**34161.2166**

In order to solve this problem, several commonly used loss functions as the following are considered in our study:

$$\begin{aligned} MAE &= \frac{\sum_{i=1}^{n}\left|predict_i - actual_i\right|}{n} \\ MAPE &= \frac{100\%}{n}\sum_{i=1}^{n}\frac{\left(predict_i - actual_i\right)}{actual_i} \\ MSE &= \frac{\sum_{i=1}^{n}\left(predict_i - actual_i\right)^2}{n} \\ MLSE &= \frac{\sum_{i=1}^{n}\left[\log\left(predict_i + 1\right) - \log\left(actual_i + 1\right)\right]^2}{n} \end{aligned} \tag{3}$$

Both Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) can sufficiently define the average prediction error. In this study, a new loss function as:

$$Newloss = \frac{\sum_{i=1}^{n}\left(1+\left|actual_i\right|\right)\left(predict_i - actual_i\right)^2}{n} \tag{4}$$

is proposed. As shown in Table 4, the proposed loss function can have the most robust results. Note that the Root Mean Square Error (RMSE) value on the dataset is not necessarily related to the ability of the closed-loop system to navigate autonomously. Only the proposed loss function with one camera setting can autonomously drive through the entire path three times, even though it does not reach the highest RMSE in the offline data set.

The proposed loss function is based on MSE and the correction term for the sharp turn scenario (higher steering command error). The correction term is defined as the multiplication of the original ground truth steering command and the error for punishing higher steering command errors. It is important to let network learn not only the most-represented straight driving situations but also those cases of less appearance as sharp turns. The experiment shows that it is a critical factor for stable driving in a high speed. The new loss function proposed achieve stable autonomously driving in the jungle case for 3 laps.

E. Objective Function

The performance of our driving model during training is reviewed and visualized in Figure 11. This provides useful information about the training of the model.

• Whether the model is overfitting on the training data: it can be observed for the case that the validation loss gains over time while training loss decreases.

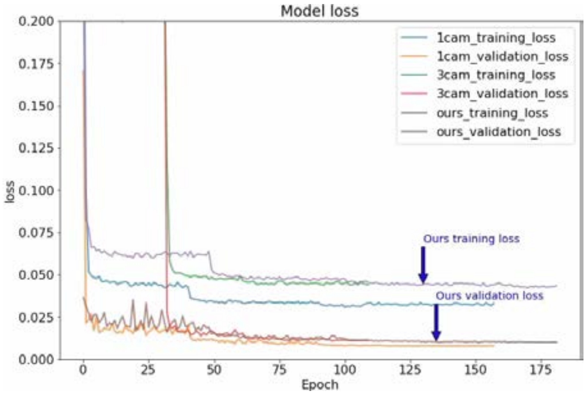


Figure 11. Model loss history.

TABLE V. GPU SERVER HARDWARE SPECIFICATIONS FOR TRAINING THE DNN

Items	**Content**
Development system	Ubuntu 16.04
Development language	Simulator: Python, Driving Platform: Pyhton, C++
Library	Keras, Tensorflow, OpenCV, Robot Operating System
CPU	Intel(R) Core(TM) i9-7900X CPU @ 3.30GHz
GPU	NVIDIA 1080 TI 11GB x 2
Memory	32 GB DDR4

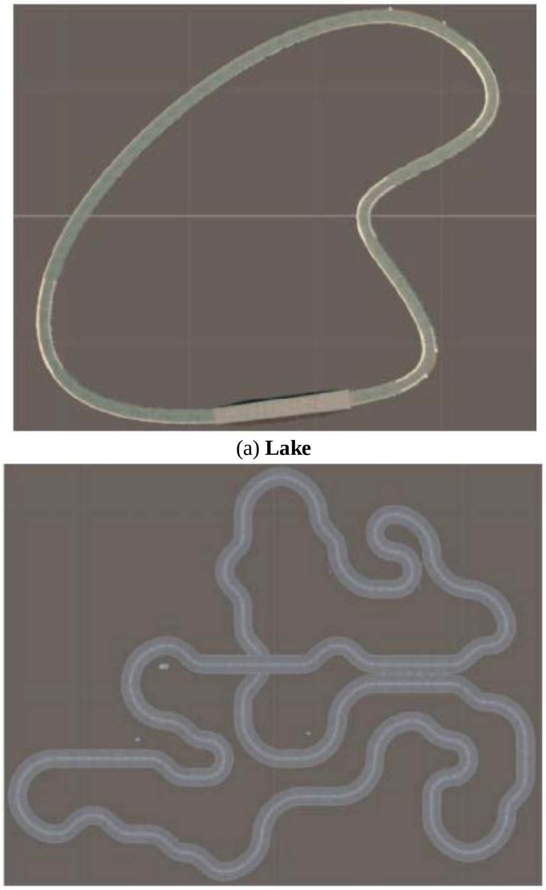
• The speed of convergence over epochs: In our cases, the model seems to converge over the first 100 epochs. As a result, a lot of training time could be saved, since there is no need for training more than 100 epochs.

As shown in the Figure 11, even though 1 camera setup has the lowest validation error, the driving network always fails on the turn in the driving simulator. This is a clear sign that the model with 1 camera did not learn how to recover from drifting. On the other hand, the model with 1 camera plus the proposed strategy is able to drive smoothly on the curve and robustly while operating in a high speed.

IV. EXPERIMENTS

In this section, to collect data, to train the neural network, and to test the driving performance for the end-to-end autonomous driving system are conducted in the Udacity open source driving simulator [6]. VisualBackProp [22] is also used to validate effectiveness of the learning methodology by visualizing learned features. The programming environment is in Python 3.5 that includes Keras library that uses TensorFlow as backend and the NVIDIA CUDA 9.0 library that supports for GPU acceleration. The training environment described in this study is based on Ubuntu 16.04 and the configuration of GPU server is shown in Table 5.

In the process, testing dataset is collected through the operation of a human expert. The pre-trained model will infer the steering command from the testing dataset and the root mean square error (RMSE) to the human expert’s steering commands is computed. The evaluation is to drive through various tracks automatically. The pre-trained model is also deployed on the driving simulator. Our model will real-time infer steer commands on the simulator. By receiving the image when driving the simulated vehicle, the model will predict steer commands and send them back to the simulator. In order to maintain the constant speed, a simple PD controller is also



(a) **Lake**

(b) **Jungle**

Figure 12: Training tracks of Udacity driving simulator. (a) the lake track is relatively easy, spacious and flat road, but the road contains a variety of different material edges. Such as racing curbs, dirt edge and bridges, (b)Jungle track is very difficult, including narrow lanes, sharp corners, and up and down steep slopes.

implemented for controlling the throttle command of the vehicle. The driving simulator will record trajectories and cross-track-error on each track for three laps **Autonomy** (how many percentage finishes in 3 laps) and **Overall CTE** (cross-track-error) are considered for evaluating the performance of the model.

Figure 12 illustrates the training course of the vehicle in the Udacity driving simulator. The car is asked to run along the course by the pre-trained model. The driving model computes the steering angle by employing the image generated with the same preprocessing method

Figures 13 and 14 shows the trajectories of the vehicle about running on those courses shown on Figure 12. The vehicle with the proposed strategy drives stably in the Lake track even at a full speed. The proposed data augmentation helps CNN learn driving in different positions on the road. In Figure 14, the model trained with a single camera failed to pass a sharp turn. In Figure 13, three cameras setup will oscillate around the lane while driving at a high speed due to lack of recovery driving data. On the contrary, the proposed methods make model successfully run along all the tracks even in unseen tracks and even driving at a high speed.

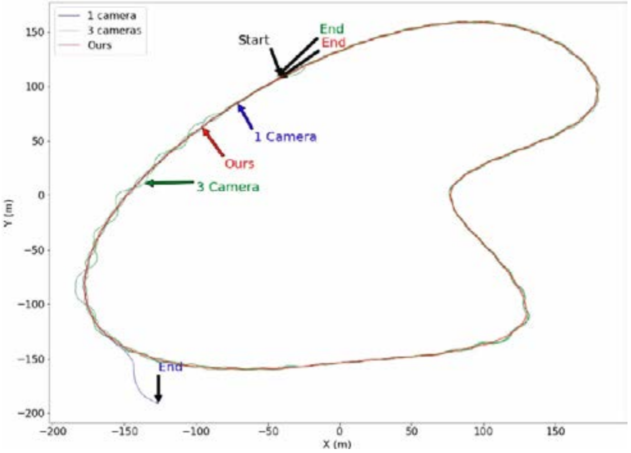


Figure 13. Autonomous driving trajectory – **Lake**.

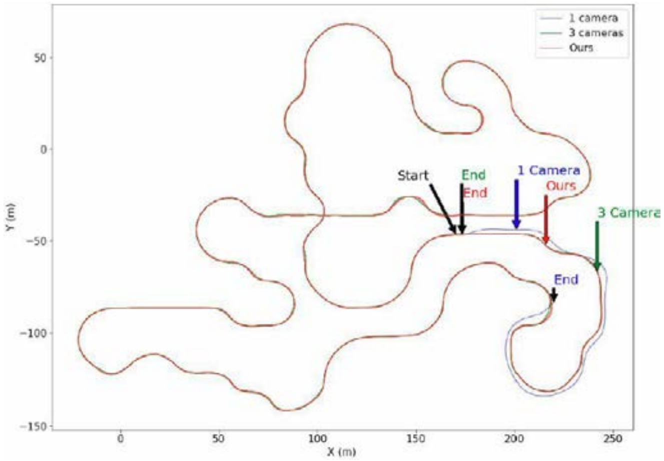
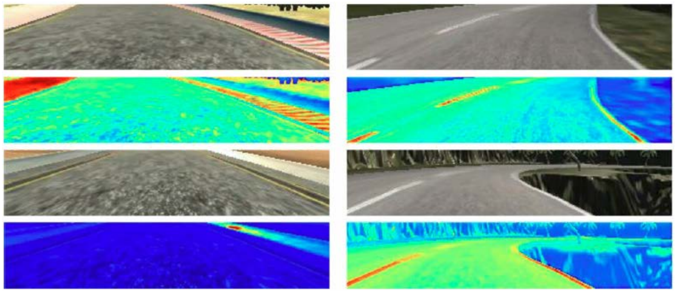


Figure 14. Autonomous driving trajectory – **Jungle**.

TABLE VI. THE OVERALL EVALUATION FOR THREE DIFFERENT APPROACHES.

Metrics	**Autonomy (3/3 laps)**		**RMSE**		**CTE**	
Track	**Lake**	**Jungle**	**Lake**	**Jungle**	**Lake**	**Jungle**
1 camera (center)	0.35/3	0.18/3	**0.1233**	**0.5332**	-	-
3 cameras (Left/Center/Right)	3/3	3/3	0.1371	0.5444	11213.104	37740.8999
1 camera with our strategy (Center + augmentation)	3/3	3/3	0.1266	0.5422	**10264.5166**	**37577.7391**

The trajectory and CTE (cross-track-error) were collected by the Udacity self- driving car simulator for the course shown in Figure 12. The cross-track-error of three different setups were compared and the accumulated errors are shown in Table 6. Three approaches are the one camera setup, 3 camera setup and the proposed strategy. The model is evaluated by the order Autonomy, CTE, and RMSE. The proposed methods and three camera setup have the same Autonomy score. As comparison of the CTE, our strategy has the lowest CTE among three different setups. The same for RMSE. Although one camera setup achieve the lowest RMSE, the driving model is overfitting the driving dataset. The driving model is not able to take a sharp turn on some simulator tracks. Three cameras setup and one camera setup with the proposed data augmentation method are able to finish 3 laps autonomously. Among them, our method can achieve the lowest CTE.



(a) **Lake** (b) **Jungle**

Figure 15. Visualizing the feature maps for (a) the lake track and (b) the Jungle track.

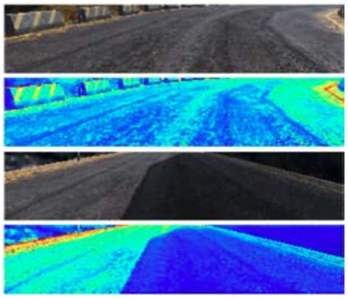


Figure 16. Visualizing the feature maps for Unseen mountain track.

In order to visualize what the driving model has learned, VisualBackProp [22] is considered to visualize what CNN see in driving videos. VisualBackProp [22] shows how the gradient flows back to image highlighting the most salient areas, as shown in Figure 15. In other words, this method can show which sets of pixels of driving image contribute most to the steering prediction. In Figure 5.8, the brighter pixel in the driving image represents the feature that the network notices, and the greater the contribution to the steering prediction. The red pixel in the driving image represents the feature that the network notices, and the greater the contribution to the steering prediction. During training process, our model is able to find the lane markers and guardrail important. In Figure 15(a), the driving model learned to notice the road features, such as curbs and lane line in **Lake** track. The same is in Figure 15 (b). In the **Jungle** track, the model is able to notice lane, guardrail and even opposite lane. More importantly, it is able to adapt to new, previously unseen environment. In Figure 16, the driving model can indeed learn the road features (road edge, lane line and red/white curbs) from **lake** and **jungle** track and then drives successfully on unseen mountain track by previously learned road features from training track.

V. Conclusions

An end-to-end learning system is introduced to learn driving policies by mimicking an expert's behavior. The deep neural network maps raw onboard observations (driving image) to steering commands. By the proposed meaningful data augmentation and a novel objective function, the system is able to perform fast off-road navigation autonomously. In the driving simulator experiments, the system is able to perform autonomous navigation even on an unseen mountain road at a top speed of 60 km/h.

In our experiments, typical data augmentation and perspective transformation from the center camera images to the simulated left and right images has shown promising ways of increasing the robustness for autonomous driving. While traditional data augmentations are very effective in image classification, these methods do not change the dataset distribution. On the other hand, the proposed method will randomly transform driving images and appropriately adjust the distribution of steer commands. To augment the dataset to a normal distribution, a kinematic bicycle model is considered to model recover controls of the augmented images. Unbalance data will cause the vehicle unstable on sharp turns since most data collected are driving straight.

The preliminary experiments in this work used the Mean Square Error (MSE) as the loss function to train the network. However, the loss function is not suitable to model less appearance situations such as the sharp turning. In our work, the proposed loss function represents better at those situations. In the driving simulator experiments, the proposed loss function can increase robustness on sharp turns. The proposed loss function is based on MSE and a weighted term of the errors on sharp turn scenarios. During the training process, the network training error on sharp turns scenarios will be greater than the error on straight driving. In this way, the driving network can learn not only the importance of driving straight but also the importance of driving on sharp turns. The results shown in our experiments are promising.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” Advances in Neural Information Processing Systems (NIPS), F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., pp. 1097–1105. Curran Associates, Inc., 2012. [Online]. Available:http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[2] B. Huval, et al, “An empirical evaluation of deep learning on highway driving,” Computing Research Repository (CoRR), vol. abs/1504.01716, 2015.

[3] F. Codevilla, et al, “End-to-end driving via conditional imitation learning,” International Conference on Robotics and Automation (ICRA), 2018.

[4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to doc-ument recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[5] P. Sermanet, et al, “Overfeat: Inte-grated recognition, localization and detection using convolutional networks,” Computing Research Repository (CoRR), vol. abs/1312.6229, 2013.

[6] Udacity, “Udacity’s self-driving car simulator.” [Online]. Available: https://github.com/ udacity/self-driving-car-sim

[7] M. Bojarski, et al, “End to end learning for self-driving cars,” Computing Research Repository (CoRR), vol. abs/1604.07316, 2016. [Online]. Available: http://arxiv.org/abs/1604.07316

[8] D. A. Pomerleau, “Alvinn: An autonomous land vehicle in a neural network,” pp.305–313, 1989. [Online]. Available: http://papers.nips.cc/paper/95-alvinn-an-autonomous-land-vehicle-in-a-neural-network.pdf

[9] A. Kendall, M. Grimes, and R. Cipolla, “PoseNet: A convolutional network for real-time 6-DOF camera relocalization,” Proceedings of the International Conference on Computer Vision (ICCV), 2015.

[10] S. Brahmbhatt, et al, “Geometry-aware learning of maps for camera localization,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[11] C. Hubschneider, et al, “Integrating end-to-end learned steering into probabilistic autonomous driving,” 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), pp. 1–7, Oct

2017.

[12] A. Carvalho, et al, “Predictive control of an autonomous ground vehicle using an iterative linearization approach,” 2013 IEEE 16th International Conference on Intelligent Transportation Systems (ITSC), pp. 2335–2340, 2013.

[13] C. E. Beal and J. C. Gerdes, “Model predictive control for vehicle stabilization at the limits of handling,” IEEE Transactions on Control Systems Technology, vol. 21, no. 4, pp. 1258– 1269, July 2013.

[14] J. Levinson, et al, “Towards fully autonomous driving: Systems and algorithms,” IEEE Intelligent Vehicles Symposium, pp. 163–168, 2011.

[15] R. Rajamani, Vehicle Dynamics and Control, ser. Mechanical Engineering Series. Springer US, 2011. [Online]. Available: https://books.google.com.tw/books?id= cZJFDox4KuUC

[16] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, “Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition,” Neural Networks, vol. 32, pp. 323–332, 2012.

[17] I. P. Alonso, et al, “Combination of feature extraction methods for SVM pedestrian detection,” IEEE Trans. Intelligent Transportation Systems, vol. 8, no. 2, pp. 292–307, 2007.

[18] A. Gurghian, et al, “Deeplanes: End-to-end lane position estimation using deep neural networks,” Computer Vision and Pattern Recognition (CVPR) Workshops, pp. 38–45. IEEE Computer Society, 2016

[19] Y. LeCun, et al, “DAVE: Autonomous off-road vehicle control using end-to-end learning,” Courant Institute/CBLL, Tech. Rep. DARPA-IPTO Final Report, 2004.

[20] G. Williams, et al, “Autonomous racing with autorally vehicles and differential games,” arXiv preprint arXiv:1707.04540, 2017

[21] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, “Kinematic and dynamic vehicle models for autonomous driving control design,” IEEE Intelligent Vehicles Symposium, pp. 1094–1099, 2015. [Online]. Available: http://dblp.uni-trier.de/db/conf/ivs/ ivs2015.html#KongPSB15

[22] M. Bojarski, et al, “Visualbackprop: visualizing cnns for autonomous driving,” Computing Research Repository (CoRR), vol. abs/1611.05418, 2016. [Online]. Available: http: //arxiv.org/abs/1611.05418

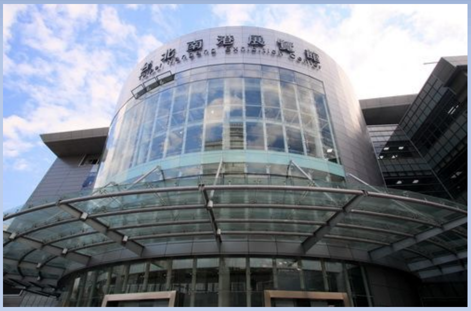
Hong-Ming Chen received the M.S. degree in the Department of Electrical Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan, in 2018. Mr. Chen also works as a research trainee for Eidgenössische Technische Hochschule Zürich (ETHZ). His research interests include deep learning, ROS, and mobile robotics.

Shun-Feng Su received the B.S. degree in electrical engineering from National Taiwan University, Taiwan, R.O.C., in 1983, and the M.S. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, Indiana, in 1989 and 1991, respectively. He is now a chair Professor in the Department of Electrical Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan. He is an IEEE Fellow and also a CACS Fellow. He has published more than 160 refereed journals and conference papers in the areas of robotics, intelligent control, fuzzy systems, neural networks, and non-derivative optimization. His current research interests include neural networks, fuzzy modeling, machine learning, virtual reality simulation, intelligent transportation systems.

ARIS2019 / NCAR2019

2019 International conference on Advanced Robotics and Intelligent Systems

2019 National Conference on Advanced Robotics



Taipei Nangang Exhibition Center, Taiwan, August 20-23, 2019

aris2019.nchu.edu.tw | ncar2019.nchu.edu.tw

ARIS2019

Host by

RST (Robotics Society of Taiwan)
NCHU (National Chung Hsing University) & ISU (I-Shou University)
TAIROA (Taiwan Automation Intelligence and Robotics Association)

Technically Sponsored by

IEEE **SMC** Society (Systems, Man and Cybernetics)

Co-Sponsored by

MOST, Taiwan, R.O.C. (Ministry of Science and Technology)
AI Robotics Hub @ **CTSP**(Central Taiwan Science Park)
CTSP (Central Taiwan Science Park)
IEEE **RA** Society, Taipei Chapter (Robotics and Automation)
CACS (Chinese Automatic Control Society)
IFSA (International Fuzzy Systems Association)
KROS (Korea Robotics Society), Korea
IEEE **SMC** Society, Taichung Chapter (Systems, Man and Cybernetics)
IEEE Computational Intelligence Society, Taipei Chapter

NCAR2019

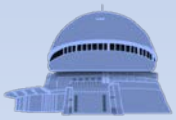
主辦單位：

台灣機器人學會 (RST)
國立中興大學 (NCHU) & 義守大學(ISU)
台灣智慧自動化與機器人協會 (TAIROA)

技術協助：

IEEE SMC學會

協辦單位：



中華民國科技部
中科智慧機器人自造基地
中部科學工業園區管理局(CTSP)
IEEE SMC學會台中分會
韓國機器人學會
IEEE 機器人與自動化學會 (RA) 台北分會
國際模糊系統學會 (IFSA)
中華民國自動控制學會 (CACS)
eTop 科技部工程科技推廣平台

Call for Paper

Topics of interest include but are not limited to
論文主題希望以下列內容為主，但並未強制限定

3D Perception	Automation Intelligence	Big-Data Analytics for Industry 4.0
Brain Machine Interface System	Cloud Computing for Industry 4.0	Cognitive Robots
Cyber-Physical Systems for Industry 4.0	Education Robots	Embedded Robot and Intelligence
Entertainment and Amusement Robots	Fuzzy Control and Robotics	Home Robots and Service Robots for Elder and Children
Human-Robot Interactions	Humanoid Robot	Indoor and Outdoor Service Robots
Inspection Robots	Intelligent Automated System and Its Applications	Intelligent Control for Assistive Robots
Intelligent Control Systems	Intelligent Manufacturing Systems	Intelligent Medical Systems and Applications
Intelligent Robotics and Systems	Intelligent Systems for Smart City	Internet of Things (IoT) for Industry 4.0
Localization and SLAM	Medical Robots	Multi-Agent Systems
Multi-Robot Systems	Rehabilitation Robot & Caring Robot	Rescue Robots and Hazardous Duty Robots
Security Robots	System Integration Technology	Wheelchair Robots
Wireless and Intelligent Sensor Fusion	Other Robotic Technologies	Other Robot-related Automation .

ARIS2019 Contributed Papers:

[Regular paper]
Agree to be included in IEEE Xplore.
[Position paper]
Excluded in IEEE Xplore.
[Presentation-Only paper]
Excluded in IEEE Xplore.

Important Dates /重要日期

May 10, 2019.	**Proposals of organized/invited sessions**
May 31, 2019.	**Submission of contributed papers**
June 28, 2019.	**Notification of acceptance**
July 26, 2019.	**Final papers due**

議程提案截止	2019/05/10
論文投稿截止	2019/05/31
論文接受通知	2019/06/28
論文完稿截止	2019/07/26

Contest Information / 競賽資訊

ARIS2019
Best Conference/Presentation/Student Paper Contest:

Select your interesting paper contest(s) while uploading your manuscript electronically.

NCAR2019
最佳研討會/學生論文競賽：

大會將根據投稿論文與學生論文內容與現場發表之表現，評選優秀論文若干篇，頒發獎狀與獎金以茲鼓勵。

ARIS2019 Conference Contact Information:

Web site: http://aris2019.nchu.edu.tw
Secretariat: *Advanced Electrical Control Lab. @ National Chung-Hsing University (NCHU)*
Phone: +886-4-22851549 ext. 601
Fax: +886-4-22856232
Address: Department of Electrical Engineering, 145, XingDa Road, Taichung 40227, Taiwan, R.O.C.
E-mail: aris2019@nchu.edu.tw

NCAR2019 研討會聯絡資訊：

研討會網址：http://ncar2019.nchu.edu.tw
秘書處：國立中興大學　尖端電控實驗室
電話：04-2285-1549#601
傳真：04-2285-6232
地址：(40227)臺中市南區興大路145號電機大樓601室
電子郵件：ncar2019@nchu.edu.tw

Information for Authors

Aim and Scope

The *i***Robotics** is an official journal of Robotics Society of Taiwan (RST) and is published quarterly. The *i***Robotics** will consider high quality papers that deal with the theory, design, and application of intelligent robotic system, intelligent artificial system, and extension theory systems ranging from hardware to software. Survey and expository submissions are also welcome. Submission of a manuscript should indicate that it has been neither published, nor copyrighted, submitted, accepted for publication elsewhere (except that the short version may have been presented at the conferences). Submitted manuscripts must be typewritten in English and as concise as possible.

Process for Submission of a Manuscript

The *i***Robotics** publishes two types of articles: regular papers and technical notes. All contributions are handled in the same procedure, where each submission is reviewed by an associate editor who makes an initial decision to send the manuscript out for peer review or to reject without external review. Articles can be rejected at this stage for a variety of reasons, such as lack of novelty, topics outside the scope of the Journal, flaws in the scientific validity, or unprofessional presentation. We are therefore not normally able to provide authors with feedback on rejected manuscripts. If the associate editor believes the article may be of interest to our readers, it is then sent out for external peer review by at least two external reviewers. According the recommendation of the associate editor, the Editor-in-Chief makes the final decision. All manuscripts should be submitted electronically in Portable Document Format (PDF) through the manuscript submission system at [http://www.rst.org.tw]. The corresponding author will be responsible for making page proof and signing off for printing on behalf of other co-authors. Upon acceptance of a paper, authors will be requested to supply their biographies (100 to 200 words) and two copies of the final version of their manuscript (in DOC format and in PDF format).

Style for Manuscript

Papers should be arranged in the following order of presentation:

1) First page must contain: a) Title of Paper (without Symbols), b) Author(s) and affiliation(s), c) Abstract (not exceeding 150 words for Papers or 75 words for Technical Note, and without equations, references, or footnotes), d) 4-6 suggested keywords, e) Complete mailing address, email address, and if available, facsimile (fax) number of each author, f) Preferred address for correspondence and return of proofs, and g) Footnotes (if desired).
2) The text: Submitted manuscripts must be typewritten double-spaced. All submitted manuscripts should be as concise as possible. Regular papers are normally limited to 26 double-spaced, typed pages, and technical notes are 12 double-spaced, typed pages. Please see the Page charge for those who want to submit long papers.
3) Acknowledgements of financial or other support (if any).
4) References: References should be numbered and appear in a separate bibliography at the end of the paper. Use numerals in square brackets to cite references, e.g., [15]. References should be complete and in the style as follows.
   [1] C. C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller - Part I," *IEEE Trans. Syst. Man Cybern.*, vol. 20, no. 2, pp. 404-418, 1990.
   [2] C. Golaszewski and P. Ramadge, "Control of discrete event processes with forced events," in *Proc. of 26th IEEE Conf. Decision and Control*, Los Angeles, CA, pp. 247-251, Dec. 1987.
   [3] P. E. Wellstead and M. B. Zarrop, *Self-Tuning Systems*, New York: Wiley, 1991.
   [4] Project Rezero, available at http://rezero.ethz.ch/project_en.html (last visited: 2017-07).
5) Tables
6) Captions of figures (on separate sheet of paper)

Style for Illustrations

1) It is in the author's interest to submit professional quality illustrations. Drafting or art service cannot be provided by the Publisher.
2) Original drawings should be in black ink on white background. Maximum size is restricted to 17.4 by 24.7 cm. Glossy prints of illustrations are also acceptable.
3) All lettering should be large enough to permit legible reduction of the figure to column width, sometimes as small as one quarter of the original size. Typed lettering is usually not acceptable on figures.
4) Provide a separate sheet listing all figure captions, in proper style for the typesetter, e.g., "Fig. 5. The error for the proposed controller."
5) Illustrations should not be sent until requested, but authors should be ready to submit these immediately upon acceptance for publication.

Page Charges

After a manuscript has been accepted for publication, the author's company or institution will be approached with a request to pay a page charge to cover part of the cost of publication. The charges include:

1) NT$ 5000 for the 10 printed pages of a full paper or for the 6 printed pages of a short paper, and the excess page charge of NT$ 1500 per extra printed page for both full and short papers.
2) For color figures or tables, an additional cost will be charged. The cost, depending on the number of color figures/tables and the final editing result, will be given upon the acceptance of this paper for publication.

Copyright

It is the policy of the RST to own the copyright of the technical contributions. It publishes on behalf of the interests of the RST, its authors, and their employers, and to facilitate the appropriate reuse of this material by others. Authors are required to sign a RST Copyright Form before publication.

Manuscripts (in PDF Format) Submission Website: http://www.rst.org.tw

iRobotics

VOLUME 2, NUMBER 1 MARCH, 2019

CONTENTS

REGULAR PAPERS

TECHNICAL NOTE

iRobotics

Volume 2, Number 1, MARCH, 2019