Experimental validation of an autonomous control system on a mobile robot platform

W. Ren, J.-S. Sun, R. Beard and T. McLain

Abstract: An autonomous control system designed for a non-holonomic wheeled mobile robot that is programmed to emulate a fixed-wing unmanned air vehicle (UAV) flying at constant altitude is experimentally validated. The overall system is capable of waypoint navigation, threat avoidance, real-time trajectory generation and trajectory tracking. Both the wheeled mobile robot experimental platform and the hierarchical autonomous control software architecture are introduced. Programmed to emulate a fixed-wing UAV flying at constant altitude, a non-holonomic mobile robot is assigned to follow a desired time-parameterised trajectory generated by a real-time trajectory generator to transition through a sequence of targets in the presence of static and popup threats. Hardware results of the autonomous control system where the trajectory tracker applies two velocity controllers accounting for fixed-wing UAV-like input constraints, are compared to simulation results of dynamic controllers that are based on non-smooth backstepping to demonstrate the effectiveness of the overall system.

1 Introduction

The development of fully autonomous control systems for robotic vehicles is made possible due to the increasing power of computational resources. Robotic vehicles are expected to find potential applications in military operations, search and rescue, environment monitoring, commercial cleaning, material handling and homeland security.

One of the challenges inherent in controlling wheeled mobile robots comes from the non-holonomic constraints, to which many wheeled mobile robots are subjected. Another challenge results from actuator saturation constraints. Previous approaches to control of wheeled mobile robots can be roughly categorised as either stabilisation or tracking. Based on Brockett's necessary condition for feedback stabilisation [1] non-holonomic systems cannot be stabilised via smooth time-invariant state feedback. The stabilisation of wheeled mobile robots is mainly tackled via discontinuous feedback [2, 3] and continuous time-varying feedback [4–6]. The tracking control problem can be achieved via sliding mode approaches [7] backstepping techniques [8–10] or optimal control strategies [11].

Fully-automating unmanned air vehicles (UAVs) poses both theoretical and practical challenges [12]. One important avenue of research is UAV path planning [13, 14]. Another important aspect focuses on trajectory optimisation for UAVs [15, 16]. In addition, effective trajectory tracking algorithms guarantee that a UAV can accurately follow its pre-specified time-parameterised desired trajectory [17].

UAVs equipped with low-level altitude-hold, velocity-hold and heading-hold autopilots can be modelled by kinematic equations similar to those of wheeled mobile robots. The inherent properties of fixed-wing UAVs impose input constraints of positive minimum velocity due to the stall conditions of the aircraft, bounded maximum velocity due to thrust limitations and saturated heading rate due to roll angle and pitch-rate constraints. However, most existing approaches to wheeled mobile robot tracking control are not directly applicable to the UAV problem since negative velocity commands are allowed in these approaches.

Although autonomous control systems have been studied extensively in theory, experimental implementation and demonstration on hardware platforms remain a challenge. The main contribution of this paper is to experimentally implement and validate an autonomous control system design procedure on a wheeled mobile robot platform that is programmed to emulate a miniature fixed-wing UAV flying at constant altitude as a proof of concept and as a step towards, hardware tests on miniature fixed-wing UAVs. In particular, velocity and heading rate constraints similar to those of UAVs are imposed on the wheeled mobile robot. The autonomous control system is capable of waypoint navigation, threat avoidance, real-time trajectory generation and trajectory tracking. Although its ability to fully emulate a fixed-wing UAV is limited, a wheeled mobile robot platform provides an effective tool to pre-test algorithms designed for UAVs. From our past experience, the methodology of first testing some UAV algorithms on the wheeled mobile robot platform and then implementing them on UAV hardware not only helps us learn practical implementation issues but avoids damaging UAV crashes. It is worthwhile to mention that although we use a wheeled mobile robot for a case study of an autonomous control system design, the same design principle can be applied or extended to other kinds of robotic vehicles.

The experimental results presented in this paper are related to the simulation studies in [17]. In this paper, we conduct experimental tests where a non-holonomic mobile robot is assigned to follow a desired trajectory generated by a real-time trajectory generator to transition through

doi:10.1049/iet-cta:20070017

Paper first received 20th January and in revised form 16th April 2007

W. Ren is with the Department of Electrical and Computer Engineering, Utah State University, Logan, UT 84322-4120, USA

J.-S. Sun, R. Beard and T. McLain are with Brigham Young University, Provo, UT 84602, USA

E-mail: wren@engineering.usu.edu

several targets in the presence of static and popup threats. Hardware results of the autonomous control system where the trajectory tracker is based on a saturation velocity controller and a discontinuous velocity controller following the design in [17] are presented. These hardware results are also compared to simulation results of the autonomous control system where the trajectory tracker uses two novel dynamic controllers based on non-smooth backstepping. The controllers in this paper are computationally simple and can be easily implemented with low-cost microcontrollers onboard vehicles. In addition, the controllers are shown to be robust to measurement noise in an experimental setting. Finally, only piecewise continuity is required for the reference velocities while other approaches for mobile robot tracking control require uniform continuity [8–10]. This feature makes the controllers feasible for our system, where the real-time trajectory generator outputs 'bang-bang' control references to achieve time-extremal properties [16].

2 Non-holonomic mobile robot experimental platform

2.1 Robot equations of motion

The kinematic equations of a non-holonomic wheeled mobile robot are given by

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

where (x, y) is the Cartesian position of the robot centre, ψ is the orientation, v is the linear velocity and ω is the angular velocity. The simplified dynamic equations of motion are given by

$$m\dot{v} = F, \quad J\dot{\omega} = \tau \tag{2}$$

where m is the mass, J is the mass moment of inertia, F is the force and τ is the torque applied to the robot. Here friction effects have been neglected.

UAVs equipped with low-level altitude-hold velocity-hold and heading-hold autopilots can be modelled by kinematic equations similar to those of wheeled mobile robots (see [17] and references therein). The inherent properties of fixed-wing UAVs impose input constraints of positive minimum velocity due to the stall conditions of the aircraft, bounded maximum velocity due to thrust limitations and saturated heading rate due to roll angle and pitch-rate constraints. In order to emulate a fixed-wing UAV flying at constant altitude, the following input constraints are imposed on the wheeled mobile robot

$$\mathcal{U}_1 = \{0 < v_{\min} \le v \le v_{\max}, -\omega_{\max} \le \omega \le \omega_{\max}\} \tag{3}$$

where $\omega_{\max} > 0$.

2.2 Robot hardware

Fig. 1*a* shows the canister robot used in our experiments. Velocity and angular velocity control commands from a host computer are sent to a PC/104 computer onboard the mobile robot over a wireless LAN. The PC/104 stack shown in Fig. 1*b* then sends motor commands to and receives encoder count updates from the multiple agent intelligence coordination and control (MAGICC) board. The MAGICC board, as shown in Fig. 1*c*, is a re-configurable high performance hardware controller designed for use in light weight, compact robotics and control applications at BYU [18]. The MAGICC board combines a 29 MHZ Rabbit processor with six high current motor drivers, four hardware quadrature turn encoders and ten analog inputs. The MAGICC board produces PWM output to the motors and calculates the robot's linear and angular velocities using encoder data. The robot's position and orientation are measured by an overhead camera.

3 Autonomous control system architecture

We implement a hierarchical software architecture in our experiments. As shown in Fig. 2, the architecture consists of five components: waypoint path planner, real-time trajectory generator, trajectory tracker, low-level robot control and the physical vehicle. The waypoint path planner and trajectory generator have been addressed in [14, 16, 19, 20]. In the following, we overview each layer.

3.1 Waypoint path planner

The waypoint path planner generates waypoint paths (straight-line segments) that change in accordance with the dynamic environment consisting of the location of the robot, the targets and the static or popup threats. The waypoint planning technique is based on the construction and search of a Voronoi graph or its extension [21], which is then searched via Eppstein's k-best paths algorithms [22]. The generated waypoint path does not account for the dynamic constraints of the vehicle so that the path search space can be significantly reduced. By using threat locations to construct the graph, the planned path accounts for the static and popup threats in the sense that the resulting straight-line segments are equidistant from the closest threats. In our paper, the Voronoi diagram is formulated based on the idea of a point threat, which represents the location of a ground-based radar. In searching the diagram to find the best path, each edge of the Voronoi diagram is assigned two costs: a threat cost and a length cost. The threat cost is based on $1/d^4$, where d is the distance from the path to the radar location [19]. If desired, the Voronoi approach can also be applied to directly account for 2-D obstacles. For example, if the 2-D obstacles are polygons, then a simple approach is to use the corners of the polygons to create the Voronoi diagram. The edges that intersect obstacles are then discarded to produce a candidate graph. Note that when a popup threat is detected or a new target occurs, the waypoint path planner will replan the path for the robot. Fig. 3 shows a typical waypoint path generated by the waypoint path planner, where the threat locations to be avoided are represented by dots, and the waypoint path is shown in solid line.

3.2 Trajectory generator

The real-time trajectory generator smoothes through the corners of the waypoint paths so that the fixed-wing UAV-like dynamic constraints (e.g. minimum turn radius) can be accounted for. The trajectory generator produces a feasible time-parameterised desired trajectory, that is, the desired position $(x_r(t), y_r(t))$ and orientation $\psi_r(t)$ for the robot. The desired reference trajectory $(x_r, y_r, \psi_r, v_r, \omega_r)$ generated by the trajectory generator satisfies

$$\dot{x}_r = v_r\cos(\psi_r), \quad \dot{y}_r = v_r\sin(\psi_r), \quad \dot{\psi}_r = \omega_r$$

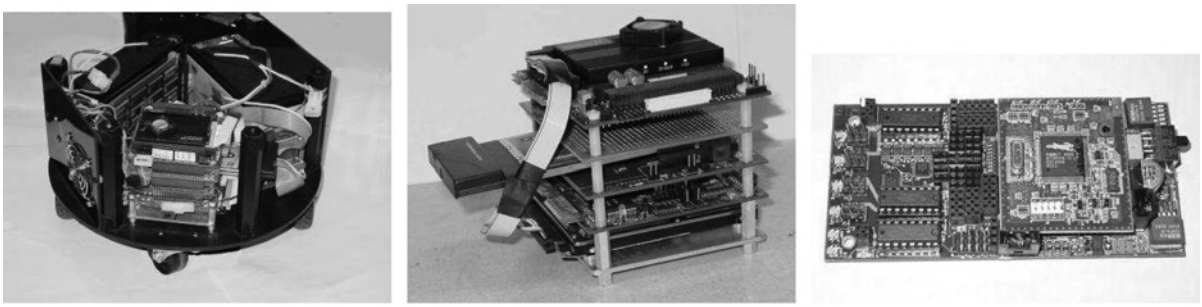

Fig. 1 *Robot hardware*
a Canister robot
b PC/104 stack
c MAGICC board

where v_r and ω_r are piecewise continuous and satisfy $\inf_{t\geq 0} v_r(t) > v_{\min} > 0$, $\sup_{t\geq 0} v_r(t) > v_{\max}$, and $\sup_{t\geq 0}|\omega_r(t)| > \omega_{\max}$. Note that the desired trajectory takes into account the UAV-like velocity and heading rate constraints.

Without loss of generality, the constraints for v_r and ω_r can be written as

$$\begin{aligned} v_{\min} + \epsilon_{v1} \leq v_r \leq v_{\max} - \epsilon_{v2} \\ -\omega_{\max} + \epsilon_{\omega 1} \leq \omega_r \leq \omega_{\max} - \epsilon_{\omega 2} \end{aligned} \tag{4}$$

where ϵ_{v1}, ϵ_{v2}, $\epsilon_{\omega 1}$ and $\epsilon_{\omega 2}$ are positive control parameters. The inclusion of ϵ_* in (4) is to guarantee that there is sufficient control authority to track the reference trajectory.

3.3 Trajectory tracker

The trajectory tracker outputs the velocity command v^c and angular velocity command ω^c to the low-level control layer so that the robot can accurately follow the desired trajectory. With regard to the kinematic model (1), the control objective is to find feasible inputs v and ω such that $|x_r - x_r - x| + |y_r - y| + |\psi_r - \psi| \to 0$ as $t \to \infty$.

Transforming the tracking errors expressed in the inertial frame to the robot frame, the error coordinates become

$$\begin{bmatrix} x_e \\ y_e \\ \psi_e \end{bmatrix} = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \psi_r - \psi \end{bmatrix}$$

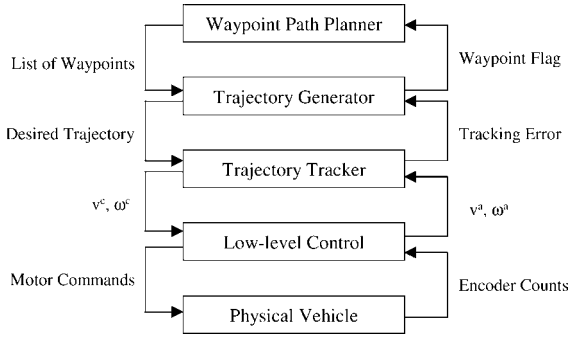

Fig. 2 *Software architecture*

Accordingly, the tracking error model can be represented as

$$\underbrace{\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\psi}_e \end{bmatrix}}_{\dot{\chi}} = \underbrace{\begin{bmatrix} \omega_r y_e \\ -\omega_r x_e + v_r \sin(\psi_e) \\ 0 \end{bmatrix}}_{f_1(t,\chi)} + \underbrace{\begin{bmatrix} -y_e & -1 \\ x_e & 0 \\ 1 & 0 \end{bmatrix}}_{g_1(\chi)} \underbrace{\begin{bmatrix} u_\omega \\ u_v \end{bmatrix}}_{u} \tag{5}$$

where $\chi = [x_e,\ y_e,\ \psi_e]^{\mathrm{T}}$, $u_\omega = \omega_r - \omega^c$ and $u_v = v^c - v_r \cos(\psi_e)$. Note that the velocity and angular velocity commands to the low-level control layer are

$$\begin{aligned} v^c &= v_r \cos(\psi_e) + u_v \\ \omega^c &= \omega_r - u_\omega \end{aligned}$$

The input constraints under the transformation become

$$\mathcal{U}_2 = \{u_\omega, u_v | \underline{\omega} \leq u_\omega \leq \bar{\omega},\ \underline{v} \leq u_v \leq \bar{v}\} \tag{6}$$

where $\underline{\omega} \triangleq \omega_r - \omega_{\max}$, $\bar{\omega} \triangleq \omega_r + \omega_{\max}$, $\underline{v} \triangleq v_{\min} - v_r \cos(\psi_e)$, and $\bar{v} \triangleq v_{\max} - v_r \cos(\psi_e)$ are time-varying due to state dependence and time-varying properties of v_r and ω_r.

Let $k > 1/2$, $\gamma_0 > 0$, $\gamma_1 \in (0, 1)$ and $\gamma_2 \in (0, 1)$ be constant. Also let

$$\kappa = \max\left\{M_0,\ 1 + \frac{d_2}{\min\{\epsilon_{\omega 1}, \epsilon_{\omega 2}\}}\right\}$$

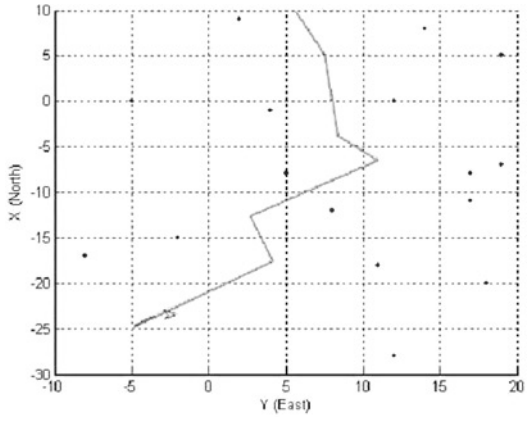

Fig. 3 *Typical waypoint path generated by the waypoint path planner*

where

$$M_0 \triangleq \max\left\{\frac{2}{\cos^{-1}\left(\frac{v_{\min}}{v_{\min}+\min\{\epsilon_{v1},\,\epsilon_{v2}\}}\right)},\,1\right\}$$
$$+\sqrt{2}\frac{d_1}{\min\{\epsilon_{\omega 1},\,\epsilon_{\omega 2}\}}\Bigg\}$$

$$d_1 \triangleq \left(k+\frac{1}{2}\right)\left[2v_{\max}-\min\{\epsilon_{v1},\,\epsilon_{v2}\}\right]$$
$$+\gamma_2\left(k-\frac{1}{2}\right)\min\{\epsilon_{v1},\,\epsilon_{v2}\}$$
$$+k[(v_{\max}-\min\{\epsilon_{v1},\,\epsilon_{v2}\})$$
$$+\gamma_1(v_{\min}+\min\{\epsilon_{v1},\,\epsilon_{v2}\})]$$
$$+(\omega_{\max}-\min\{\epsilon_{\omega 1},\,\epsilon_{\omega 2}\})$$
$$+(v_{\max}-\min\{\epsilon_{v1},\,\epsilon_{v2}\})+\gamma_0$$

$$d_2 \triangleq (v_{\max}-\min\{\epsilon_{v1},\,\epsilon_{v2}\})$$
$$\times\left[\sqrt{2}\left(k-\frac{1}{2}\right)\frac{M_2}{M_0}+\sqrt{2}k\frac{M_1}{M_0}+1\right]$$
$$+(\omega_{\max}-\min\{\epsilon_{\omega 1},\,\epsilon_{\omega 2}\})+\gamma_0$$

and

$$M_1 \triangleq \sup_{\substack{0<|\alpha|<1/M_0\\ |\beta|<1/M_0}}\left|\frac{\sin(\alpha-\beta)+\sin(\beta)}{\alpha}\right|$$

$$M_2 \triangleq \sup_{\substack{0<|\alpha|<1/M_0\\ |\beta|<1/M_0}}\left|\frac{\cos(\beta)-\cos(\alpha-\beta)}{\alpha}\right|$$

In [17], we have shown that for $\lambda > \kappa$, where κ is defined in (7)

$$V_0(\chi) = \sqrt{\left(\lambda\psi_e+\frac{y_e}{\sqrt{x_e^2+y_e^2+1}}\right)^2+1}$$
$$+k\sqrt{x_e^2+y_e^2+1}-(1+k) \qquad (8)$$

is a constrained control Lyapunov function (CLF) for the system (5) with the input constraints (6) such that $\inf_{u\in\mathcal{U}_2}\dot{V}_0(\chi) \le -W(\chi)$, where $W(\chi)$ is a continuous positive-definite function.

In the following, we first design velocity controllers based on the kinematic model (1), where a saturation controller and a discontinuous controller will be given. Then we apply the non-smooth backstepping approach proposed in [23] to design force and torque controllers based on the dynamic model (2) for comparison purposes.

Motivated by [24], we define a feasible control set as

$$\mathcal{F}(t,\chi) = \{u\in\mathcal{U}_2 | L_fV + L_gVu \le -W(\chi)\}$$

where $L_fV = (\partial V_0/\partial\chi)f_1(t,\chi)$, $L_gV = (\partial V_0/\partial\chi)g_1(\chi)$, $V_0(\chi)$ is the constrained CLF given by (8), and $W(\chi)$ is the continuous positive-definite function such that $\inf_{u\in\mathcal{U}_2}\dot{V}_0(\chi) \le -W(\chi)$. Note that $\mathcal{F}(t,\chi)$ denotes the stabilising controls at time t (with respect to $V_0(\chi)$) that also satisfy the input constraints (6). Also note that the fact that V_0 is a constrained CLF for (5) with the input constraints (6) guarantees that $\mathcal{F}(t,\chi)$ is non-empty for any t and χ. It is worthwhile to mention that as ϵ_* approach zero, the feasible control set vanishes.

Fig. 4 shows the feasible control set at a certain time t. The line denoted by $L_gV_u + L_fV + W = 0$ separates the 2-D control space into two halves, where the left half in Fig. 4 represent the unconstrained stabilising controls satisfying $\dot{V} \le -W(\chi)$ at time t. The rectangle area denotes the time-varying input constraints (6). The shaded area represents the feasible control set $\mathcal{F}(t,\chi)$.

Define a signum-like function as $\mathrm{sgn}(a,\,b,\,c) = b$ if $a < 0$, $\mathrm{sgn}(a,\,b,\,c) = 0$ if $a = 0$ and $\mathrm{sgn}(a,\,b,\,c) = c$ if $a > 0$. By mimicking the proof in [17] that V_0 is a constrained CLF for (5), it is straightforward to verify that V_0 is a constrained Lyapunov function for (5) with control input $u_d = [u_\omega,\,u_v]^T$, where

$$u_\omega = \mathrm{sgn}(-\sigma_\omega,\underline{\omega},\bar{\omega})$$
$$u_v = \mathrm{sgn}(x_e,\underline{v},\bar{v})$$

where $\sigma_\omega \triangleq \lambda\psi_e + (y_e/(\sqrt{x_e^2+y_e^2+1}))$. Noting that $v = v_r\cos(\psi_e)+u_v$ and $\omega = \omega_r - u_\omega$, a discontinuous controller for the kinematic model (1) is given by

$$v^c = \begin{cases} v_{\min}, & x_e < 0\\ v_r, & x_e = 0\\ v_{\max}, & x_e > 0\end{cases}, \quad \omega^c = \begin{cases} \omega_{\max}, & \sigma_\omega > 0\\ \omega_r, & \sigma_\omega = 0\\ -\omega_{\max}, & \sigma_\omega < 0\end{cases}$$

Define a saturation function as $\mathrm{sat}(a,\,b,\,c) = b$ if $a < b$, $\mathrm{sat}(a,\,b,\,c) = a$ if $b \le a \le c$ and $\mathrm{sat}(a,\,b,\,c) = c$ if $a > c$, where it is assumed that $b < c$. Similarly, V_0 is also a constrained Lyapunov function for (5) with control input $u_s = [u_\omega,\,u_v]$, where

$$u_\omega = \mathrm{sat}(-\eta_\omega\sigma_\omega,\underline{\omega},\bar{\omega})$$
$$u_v = \mathrm{sat}(\eta_v x_e,\underline{v},\bar{v})$$

where η_ω and η_v are required to be greater than some positive constants which are expressed precisely in [17]. Therefore a saturation controller for the kinematic model (1) is given by

$$v^c = \begin{cases} v_{\min}, & \eta_v x_e < \underline{v}\\ v_r\cos(\psi_e)+\eta_v x_e, & \underline{v} \le \eta_v x_e \le \bar{v}\\ v_{\max}, & \eta_v x_e > \bar{v}\end{cases}$$

$$\omega^c = \begin{cases} \omega_{\max}, & -\eta_\omega\sigma_\omega < \underline{\omega}\\ \omega_r+\eta_\omega\sigma_\omega, & \underline{\omega} \le -\eta_\omega\sigma_\omega \le \bar{\omega}\\ -\omega_{\max}, & -\eta_\omega\sigma_\omega > \bar{\omega}\end{cases}$$

Note that both u_d and u_s are in the feasible set in Fig. 4. In particular, u_d corresponds to a vertex of a corner of the feasible control set.

Dynamic control laws will be derived to compare with the kinematic control laws described earlier. Given kinematic control laws, a standard way to extend the kinematic

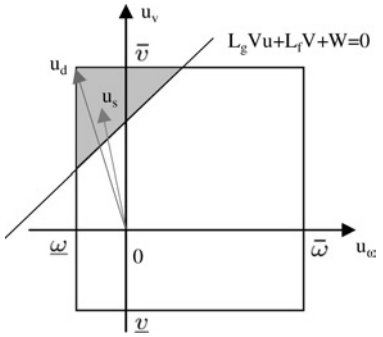

Fig. 4 *Feasible set $\mathcal{F}(t,\,\chi)$*

control laws to dynamic strategies is to apply backstepping techniques. It is obvious that both v^c and ω^c are not differentiable for the discontinuous controller and the saturation controller. Note that the continuity of the saturation controller depends on the continuity of v_r and ω_r. In this paper, v_r and ω_r are only assumed to be piecewise continuous. In fact, the reference heading rate output from the real-time trajectory generator are 'bang-bang' signals to achieve time-extremal properties. As a result, traditional backstepping techniques are not applicable to find dynamic control laws for the dynamic model (2). Therefore we resort to the non-smooth backstepping approach proposed in [23] to tackle this problem.

Note that (5) and (2) can be rewritten as

$$\dot{\chi} = f(t, \chi) + g(\chi)\xi, \quad \dot{\xi} = \nu \tag{9}$$

where $\xi = [v, \omega]^{\mathrm{T}}$, $\nu = [F/m, \tau/J]^{\mathrm{T}}$

$$f(t, \chi) = \begin{bmatrix} v_r \cos(\psi_e) \\ v_r \sin(\psi_e) \\ \omega_r \end{bmatrix}$$

and

$$g(\chi) = \begin{bmatrix} -1 & y_e \\ 0 & -x_e \\ 0 & -1 \end{bmatrix}$$

Let $\phi(t, \chi) = [v^c, \omega^c]^{\mathrm{T}}$ represent the saturation or discontinuous control law described earlier for the kinematic model (1). Let $\dot{\tilde{\phi}}(t, \chi)$ denote the generalised time derivative of ϕ and let μ represent the minimum norm element of $\dot{\tilde{\phi}}(t, \chi)$ [23, 25]. Define

$$\nu = [\nu_F, \nu_\tau]^{\mathrm{T}} = \mu - K[\xi - \phi(t, \chi)] - \left[\frac{\partial V_0}{\partial \chi} g(\chi)\right]^{\mathrm{T}} \tag{10}$$

where K is a 2×2 symmetric positive-definite matrix. Then the dynamic control law is given by

$$F = m\nu_F, \quad \tau = J\nu_\tau \tag{11}$$

Theorem 3.1: The dynamic control law (11) guarantees that $|x - x_r| + |y - y_r| + |\psi - \psi_r| + v - v_r| + |\omega - \omega_r| \rightarrow 0$ asymptotically as $t \rightarrow \infty$

Proof: Consider a Lyapunov function candidate $V = V_0(\chi) + 1/2(\xi - \phi(t, \chi))^{\mathrm{T}}(\xi - \phi(t, \chi))$. Note that $\dot{V}_0 \leq -W(\chi)$ if $\xi = \phi(t, \chi)$ from the argument that V_0 is a Lyapunov function for the kinematic model (1). Following Theorem 5 in [23], it can be verified that the control law (11) guarantees that $\|\chi\| + \|\xi - \phi(t, \chi)\| \rightarrow 0$ asymptotically as $t \rightarrow \infty$. Noting that $\phi(t, 0) = [v_r, \omega_r]^{\mathrm{T}}$, it is straightforward to see that $|x_e| + |y_e| + |\psi_e| + |v - v_r| + |\omega - \omega_e| \rightarrow 0$ asymptotically as $t \rightarrow \infty$, which in turn implies that $|x - x_r| + |y - y_r| + |\psi - \psi_r| + |v - v_r| + |\omega - \omega_r| \rightarrow 0$ asymptotically as $t \rightarrow \infty$ □

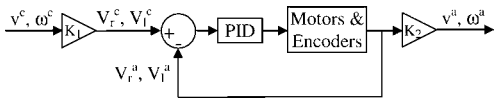

Fig. 5 *PID control loop for v^c and ω^c*

Note that unlike the case of Theorem 5 in [23], ξ does not approach zero since here we consider a tracking problem where $\phi(t, 0) = [v_r, \omega_r]^{\mathrm{T}}$ while Theorem 5 in [23] considers a stabilisation problem where $\phi(0) = 0$.

3.4 Low-level control

The low-level control layer uses the velocity and angular velocity commands to control the robot. Low-level control algorithms are implemented in the MAGICC board with the objective of maintaining commanded robot linear and angular velocities during the experiments. Fig. 5 shows a PID control loop for the commanded linear and angular velocities. Note that the trajectory tracker outputs the commanded linear and angular velocities v^c and ω^c. They are then converted to the commanded left and right wheel velocities denoted by V_l^c and V_r^c, respectively, via the conversion factor K_1. The actual left and right wheel velocities denoted V_l^a and V_r^a, respectively, are then converted back to the actual linear and angular velocities v^a and ω_a, respectively, via the conversion factor K_2.

4 Experimental results

In this section, we experimentally validate the autonomous control system design using a non-holonomic mobile robot. Hardware results based on both the saturation velocity controller and the discontinuous velocity controller derived in Section 3.3 are presented. These hardware results are also compared to simulation results using the dynamic controller (11). With a non-holonomic mobile robot programmed to emulate in hardware a UAV flying at constant altitude, the robot is assigned to follow a desired trajectory generated by the real-time trajectory generator so as to transition through several known targets in the presence of static and popup threats.

The experimental tests were conducted in the MAGICC Laboratory at Brigham Young University. The MAGICC Lab mobile robot testbed as shown in Fig. 6*a* consists of a 5×5 m^2 field. Fig. 6*b* shows the schematic hardware/software structure of the testbed.

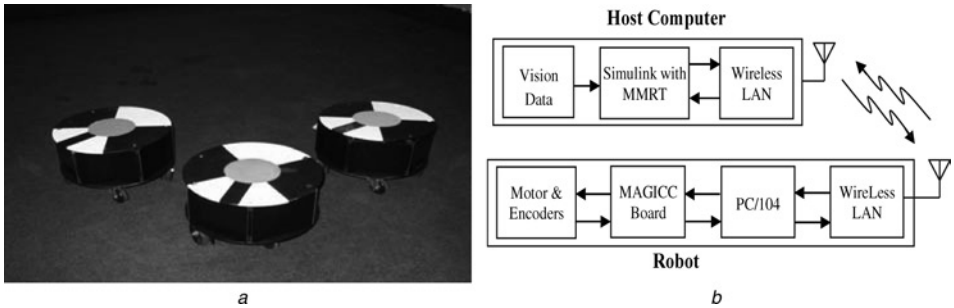

Fig. 6 *Wheeled mobile robot experimental platform*

a Mobile robot tested
b Hardware/Software structure

Table 1: Specifications of the robot and control parameters

Parameter	Value
m, kg	10.1
J, kg m^2	0.13
v_{min}, m/s	0.075
v_{max}, m/s	0.24
ω_{max}, rad/s	2
v_r, m/s	$\in$[0.15, 0.19]
ω_r, rad/s	$\in$[−1.25, 1.25]
λ	1
η_v	3
η_ω	10

In our experiments, all high-level control algorithms including the waypoint path planner, real-time trajectory generator and trajectory tracker are performed on a host computer running Matlab/Simulink under the Linux operating system. The low-level robot control algorithm is run onboard the robot. An overhead camera capturing image frames at 30 frames per second is mounted on the ceiling directly above the testbed to measure the position and orientation of each robot. The control algorithms utilise the MAGICC Mobile Robot Toolbox (MMRT) [26] that runs under Simulink. MMRT provides a convenient interface for rapid implementation and testing of mobile robot control applications. Table 1 shows the specifications of the robot and control parameters used to obtain the experimental results. Note that we have purposely imposed constraints for the linear and angular velocities of the robot such that $v \in [v_{min}, v_{max}]$ and $\omega \in [-\omega_{max}, \omega_{max}]$ to experimentally validate the results presented in Section 3. The robot emulates a target Zagi airframe-based UAV with a minimum airspeed of 8 m/s, a maximum airspeed of 13 m/s and a minimum turn radius of 11 m.

Figs. 7–9 show the hardware results of the autonomous control system where the trajectory tracker uses the two velocity controllers under relatively large control authority $|\omega^c| \leq 2$ rad/s, that is, $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.75$ rad/s. In Fig. 7, we show the trajectories of the robot transitioning through two targets in the presence of static threats and popup threats, where stars indicate the starting points of the waypoint path and the time-parameterised trajectories, dots denote the static threats, 'x' marks represent the popup threats and squares denote the targets. Note that in our experiments, we purposely introduce a large initial tracking error to test the performance of the controllers in the presence of an initial tracking error. Note that the robot consecutively reaches its targets and avoids static and popup threats by accurately following the desired time-parameterised trajectory. Fig. 8 compares the tracking errors of the two velocity controllers. Note that the position and orientation of the robot are measured by the overhead camera. As a result of vision noise from the overhead camera, there exist a steady-state tracking error of about 0.05 m and glitches in the heading tracking errors for both controllers. Note that the velocity controllers are robust to robot position and orientation measurement noise. Fig. 9 compares the reference and commanded velocities for both controllers. We can see that the velocities of the discontinuous controller switch frequently in time. Although similar performance is achieved using both controllers, we notice that the motion of the robot using the saturation controller is smooth while the motion of the robot using the discontinuous controller has significant jerks.

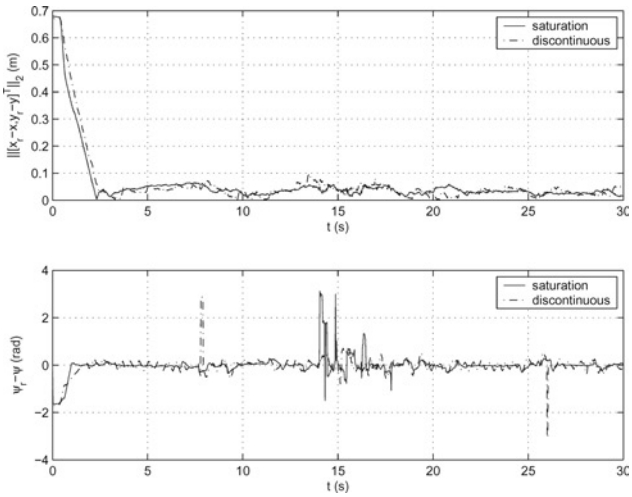

Fig. 8 *Tracking errors in an experiment using the velocity controllers when there are two targets and* $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.75$ *rad/s*

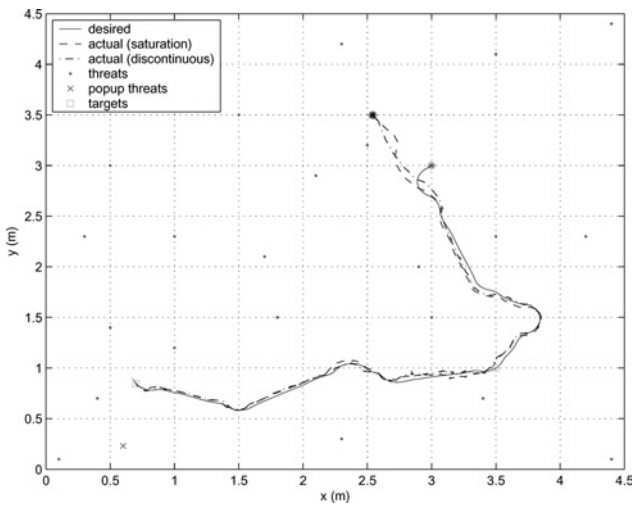

Fig. 7 *Desired and actual robot trajectories in an experiment using the velocity controllers when there are two targets and* $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.75$ *rad/s*

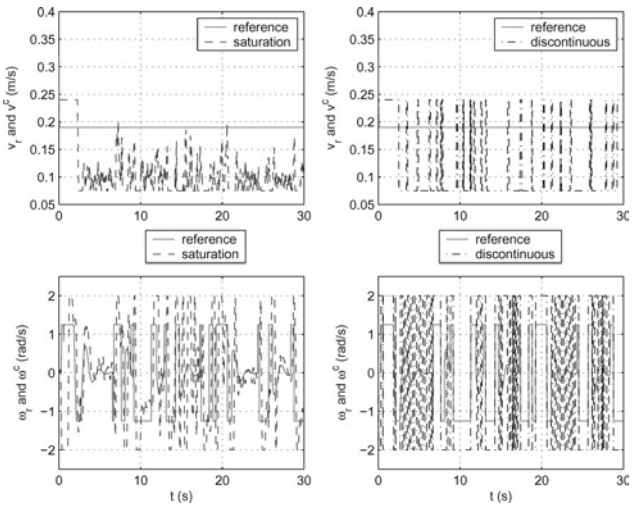

Fig. 9 *Reference and commanded velocities in an experiment using the velocity controllers when there are two targets and* $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.75$ *rad/s*

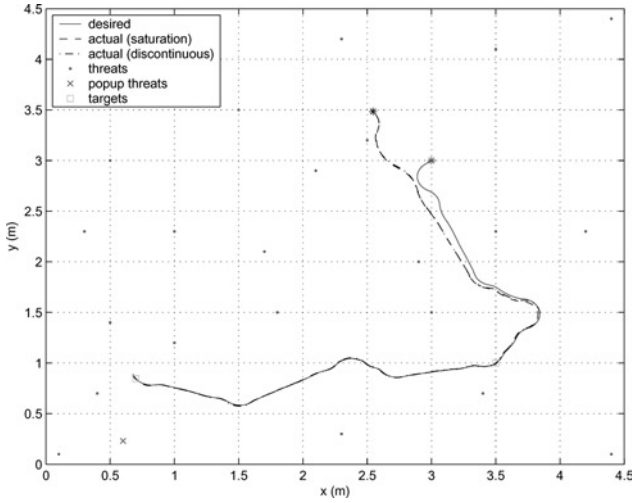

Fig. 10 *Desired and actual robot trajectories in simulation using non-smoooth backstepping when there are two targets and* $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.75\ rad/s$

In contrast to the above hardware results of the autonomous control system where the trajectory tracker uses the velocity controllers, we also show simulation results of the autonomous control system where the trajectory tracker uses the dynamic controller based on non-smooth backstepping in Figs. 10–13. Note that the dynamic controller takes into account vehicle dynamics while the velocity controllers do not. Our motivation here is to compare the experimental results based on the velocity controllers that omit vehicle dynamics in the presence of noise and uncertainties existing in an experimental setting with the ideal simulation results based on the dynamic controller. By doing so, the performance of the velocity controllers in the experimental setting can be evaluated. The robot used in our testbed has physical constraints for force and torque of $|F| \leq 30$ N and $|\tau| \leq 230$ Nm. We choose $K = \text{diag}\{2, 2\}$ in (10). Note that non-smooth backstepping is applied to both the saturation velocity controller and the discontinuous velocity controller. The dynamic controllers based on non-smooth backstepping for both velocity controllers have similar tracking performances as shown in Figs. 10 and 11. However, compared to the case using non-smooth backstepping for the saturation velocity controller, switching phenomena for actual linear and angular velocities and control forces and torques are more severe in the case

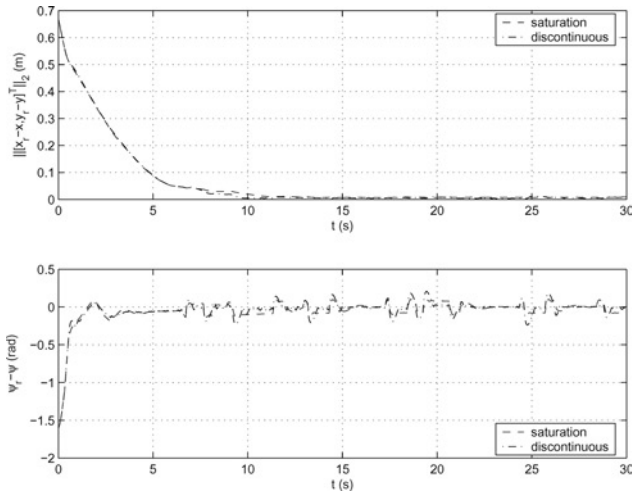

Fig. 11 *Tracking errors in simulation using non-smooth backstepping when there are two targets and* $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.75\ rad/s$

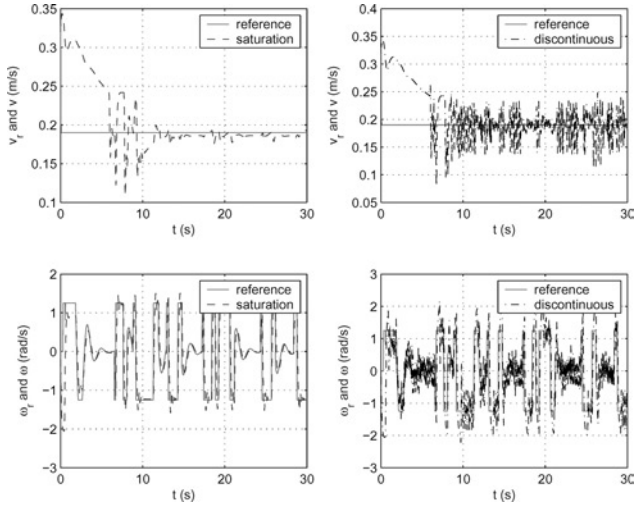

Fig. 12 *Reference and actual velocities in simulation using non-smooth backstepping when there are two targets and* $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.75\ rad/s$

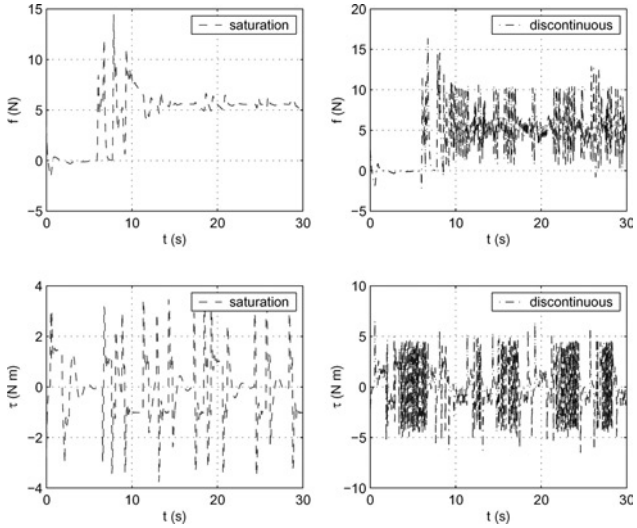

Fig. 13 *Control forces and torques in simulation using non-smooth backstepping controller when there are two targets and* $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.75\ rad/s$

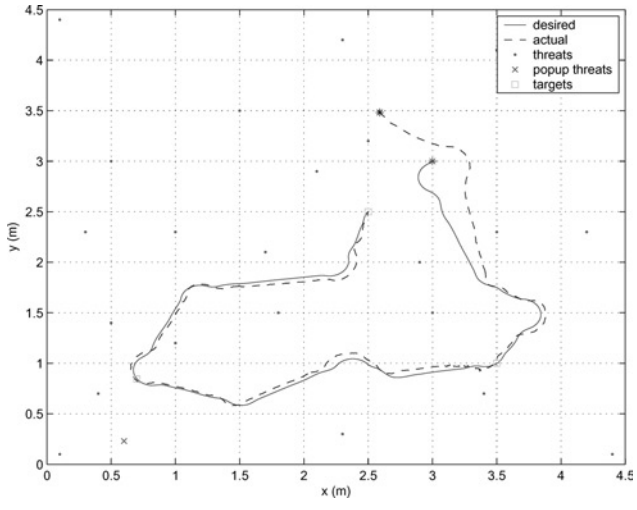

Fig. 14 *Desired and actual robot trajectories in an experiment using the saturation controller when there are three targets and* $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.2\ rad/s$

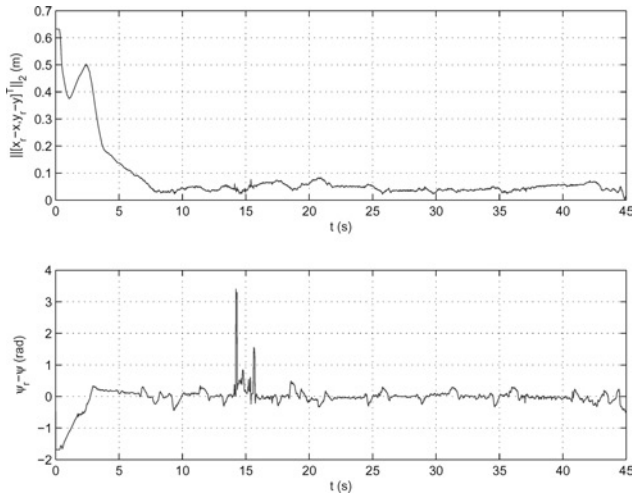

Fig. 15 *Tracking errors in an experiment using the saturation controller when there are three targets and* $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.2$ *rad/s*

of using non-smooth backstepping for the discontinuous controller as shown in Figs. 12 and 13. By comparing Figs. 8 and 11, we see that the hardware results using the velocity controllers are almost comparable to the simulation results using the dynamic controllers.

As a final test, we reduce the control authority to $|\omega^c| \leq 1.45$ rad/s, that is, $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.2$ rad/s, and introduce one more target in the test field. Figs. 14–16 show the hardware results of the autonomous control system where the trajectory tracker uses the saturation velocity controller in this situation. As shown in Fig. 16, ω^c is constrained within $[-1.45, 1.45]$ (rad/s) compared to Fig. 9 where ω^c is constrained within $[-2, 2]$ (rad/s). Note that similar performances are still achieved with much smaller control authority for ω^c and the robot transitions through three targets consecutively and avoids static and popup threats as desired.

5 Conclusion and future work

This paper has experimentally validated an autonomous control system on a wheeled mobile robot platform as a case study. The autonomous control system has the functionalities of waypoint navigation, threat avoidance, real-time trajectory generation and trajectory tracking. Programmed to emulate a UAV flying at constant altitude, a non-holonomic wheeled mobile robot was able to consecutively transition through a sequence of targets and effectively avoid static and popup threats by following a desired time-parameterised trajectory generated by a real-time trajectory generator. Future work includes implementing the whole system onboard a miniature fixed-wing UAV. There are a few implementation challenges associated with a UAV platform. One implementation challenge results from the relatively low update rate of the UAV position and heading measurement. While the main control loop runs at $\sim$130 HZ and the real-time trajectory generator runs at 20 HZ, the GPS can only provide UAV position and heading information at roughly 1 HZ. A state estimator needs to be designed to provide missing data and mitigate the low UAV position and heading update rate. Another challenge results from the fact that the autopilot response to heading step is not truly first order as assumed. In addition, explicitly accounting for tracking error due to wind is also a challenge for implementing the tracking controllers on UAV hardware.

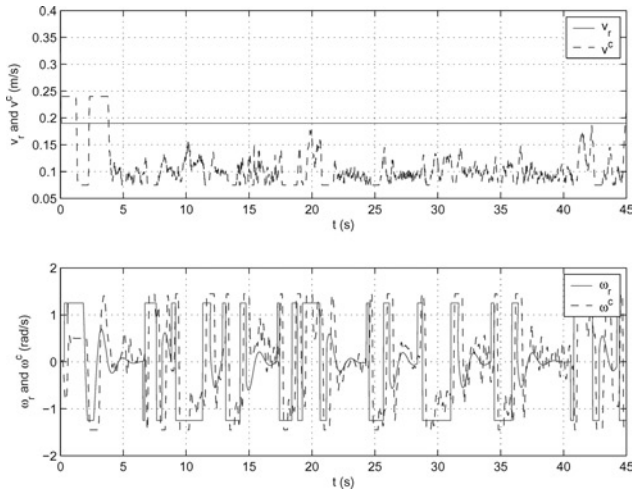

Fig. 16 *Reference and commanded velocities in an experiment using the saturation controller when there are three targets and* $\epsilon_{\omega 1} = \epsilon_{\omega 2} = 0.2$ *rad/s*

6 Acknowledgments

This paper expands on the preliminary version presented in [27]. The authors gratefully acknowledge Derek Kingston, Jeff Anderson and Matt Blake for their assistance in obtaining the experimental results.

7 Reference

1 Brockett, R.W.: 'Asymptotic stability and feedback stabilization' in Millman, R.S. and Sussman, H.J. (Eds.): 'Differential Geometric Control Theory' (Birkhaüser, 1983), pp. 181–191
2 Bloch, A.M., and McClamroch, N.H.: 'Control of mechanical systems with classical nonholonomic constraints'. Proc. IEEE Conf. on Decision and Control, Tampa, FL, December 1989, pp. 201–205
3 Astolfi, A.: 'Discontinuous control of nonholonomic systmes', *Syst. Control Lett.*, 1996, **24**, pp. 37–45
4 Pomet, J.P.: 'Explicit design of time-varying stabilizing control laws for a class of controllable systems without drift', *Syst. Control Lett.*, 1992, **18**, pp. 147–158
5 Murray, R.M., and Sastry, S.S.: 'Nonholonomic motion planning: steering using sinusoids', *IEEE Trans. Autom. Control*, 1993, **38**, (5), pp. 700–716
6 Tian, Y.-P., and Li, S.: 'Exponential stabilization of nonholonomic dynamic systems by smooth time-varying control', *Automatica*, 2002, **38**, (7), pp. 1139–1146
7 Yang, J.-M., and Kim, J.-H.: 'Sliding mode motion control of nonholonomic mobile robots', *IEEE Control Syst. Mag.*, 1999, **19**, (2), pp. 15–23
8 Jiang, Z.-P., and Nijmeijer, H.: 'Tracking control of mobile robots: A case study in backstepping', *Automatica*, 1997, **33**, pp. 1393–1399
9 Jiang, Z.-P., Lefeber, E., and Nijmeijer, H.: 'Saturated stabilization and track control of a nonholonomic mobile robot', *Syst. Control Lett.*, 2001, **42**, pp. 327–332
10 Lee, T.-C., Song, K.-T., Lee, C.-H., and Teng, C.-C.: 'Tracking control of unicycle-modeled mobile robots using a saturation feedback controller', *IEEE Trans. Control Syst. Technol.*, 2001, **9**, (2), pp. 305–318
11 Souères, P., Balluchi, A., and Bicchi, A.: 'Optimal feedback control for line tracking with a bounded-curvature vehicle', *Int. J. Control*, 2001, **74,**(10), pp. 1009–1019
12 Clough, B.T.: 'Unmanned air vehicles: autonomous control challenges, a Researcher's perspective,' in Murphey, R., and Pardalos, P.M. (Eds.): 'Cooperative control and optimization' (Kluwer Academic Publishers Series, Applied Optimization, 2002), vol. 66, pp. 35–53
13 Bortoff, S.A.: 'Path planning for UAVs'. Proc. American Control Conf, Chicago, IL, June 2000, pp. 364–368
14 Chandler, P., Rasumussen, S., and Pachter, M.: 'UAV cooperative path planning'. Proc. AIAA Guidance Navigation, and Control Conf., Denver, CO, August 2000, Paper no. AIAA-2000-4370

15 Yakimenko, O.A.: 'Direct method for rapid prototyping of near-optimal aircraft trajectories', *AIAA J. Guid. Control Dyn.*, 2000, **23**, (5), pp. 865–875
16 Anderson, E.P., Beard, R.W., and McLain, T.W.: 'Real time dynamic trajectory smoothing for uninhabited aerial vehicles', *IEEE Trans. Control Syst. Technol.*, 2005, **13**, (3), pp. 471–477
17 Ren, W., and Beard, R.W.: 'Trajectory tracking for unmanned air vehicles with velocity and heading rate constraints', *IEEE Trans. Control Syst. Technol.*, 2004, **12**, (5), pp. 706–716
18 http://www.ee.byu.edu/magic/
19 McLain, T.W., and Beard, R.W.: 'Cooperative rendezvous of multiple unmanned air vehicles'. Proc. AIAA Guidance, Navigation, and Control Conf., Denver, CO, August 2000, Paper no. AIAA-2000-4369
20 Beard, R.W., McLain, T.W., Goodrich, M., and Anderson, E.P.: 'Coordinated target assignment and intercept for unmanned air vehicles', *IEEE Trans. Robot. Autom.*, 2002, **18**, (6), pp. 911–922
21 Sedgewick, R.: 'Algorithms' (Addison-Wesley, 1988, 2nd edn.)
22 Eppstein, D.: 'Finding the k shortest paths', *SIAM J. Comput.*, 1999, **28**, (2), pp. 652–673
23 Tanner, H.G., and Kyriakopoulos, K.J.: 'Backstepping for nonsmooth systems', *Automatica*, 2003, **39**, pp. 1259–1265
24 Curtis, J.W., and Beard, R.W.: 'Satisficing: a new approach to constructive nonlinear control', *IEEE Trans. Autom. Control*, 2004, **49**, (7), pp. 1090–1102
25 Shevitz, D., and Paden, B.: 'Lyapunov stability theory of nonsmooth systems', *IEEE Trans. Autom. Control*, 1994, **39**, (9), pp. 1910–1914
26 Kelsey, J.: 'MAGICC multiple robot toolbox (MMRT): a simulink-based control and coordination toolbox for multiple robotic agents', Master's Thesis, Brigham Young University, Provo UT 84602, April 2002
27 Ren, W., Sun, J.-S., Beard, R.W., and McLain, T.W.: 'Nonlinear tracking control for nonholonomic mobile robots with input constraints: an experimental study'. Proc. American Control Conf. Portland, OR, June 2005, pp. 4985–4990