We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

186,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

20

Autonomous Robot Navigation using Flatness-based Control and Multi-Sensor Fusion

Gerasimos G. Rigatos
Unit of Industrial Automation,
Industrial Systems Institute,
26504, Rion Patras,
Greece

1. Introduction

This research work contains results on flatness-based control and sensor fusion for mobile robot systems. Flatness-based control is applicable to *differentially flat systems* i.e. to systems the behavior of which is determined by the trajectory of a finite collection of quantities, consisting of the *flat output* and its derivatives [Mounier, H. & Roudolf, J. (2001)],[Rouchon, P. (2005)]. Flatness-based control is equivalent to the feedback linearization method where the error dynamics of the closed-loop system can be described by a linear ODE after state feedback and subsequently can be stabilized using methods from linear control theory [Fliess, M. & Mounier, H. (1999)], [Roudolf, H. (2003)]. An advantage of flatness based control is that it simplifies trajectory planning and enables open-loop controller design. For linear finite dimensional systems flatness coincides with controllability, while this property can be generalized in the case of infinite dimensional systems [Laroche, B.; Martin, P. & Petit, N. (2007)], [Martin, P. & Rouchon, P. (1999)], [Meurer, T. & Zeitz, M. (2004) ,[Lévine, J. & Nguyen, D.V. (2003)].

Motion planning and control of autonomous vehicles is an important research topic in robotics (results in [Rigatos, G.G. (2003)], [Rigatos, G.G.et al., (2001)], [Rigatos, G.G. (2008)]) and flatness based control has been proposed as a suitable methodology for this class of problems [Martin, P. & Rouchon, P, (1999)]. Using the concept of flatness-based control, motion control algorithms have been developed that permit steering of the robotic vehicle along any desirable path in the 2D plane [Oriolo, G.et al., (2002)]. It will be shown that the kinematic model of the robotic vehicle is a flat system and thus can be expressed using a flat output and its derivatives. Moreover, the case in which the mobile robot's state vector is estimated through fusion of measurements from distributed sensors will be examined [Caron, F. et al., (2007)], [Jetto, L. et al., (1999)], [Yang, N. et al., (2005)]. To this end, the state vector of the robotic vehicle will be reconstructed with the use of Gaussian or non-parametric state estimators (such as Extended Kalman Filtering or Particle Filtering) [Rigatos, G.G. (2007)], [Rigatos, G.G. & Tzafestas, S.G. (2007a)], [Rigatos, G.G. (2007b)]. Simulation experiments in the case of the completely measurable state vector can show that flatness-based control enables the mobile robot to follow any reference path. Additionally, simulation experiments can show that using the state vector which is estimated from sensor

Open Access Database www.i-techonline.com

Source: Robotics, Automation and Control, Book edited by: Pavla Pecherková, Miroslav Flídr and Jindřich Duník,
ISBN 978-953-7619-18-3, pp. 494, October 2008, I-Tech, Vienna, Austria

fusion, flatness-based control is also efficient in making the mobile robot track any desirable trajectory.

The structure of the paper is as follows: In Section 2 principles of flatness-based control are analyzed and examples are given for finite dimensional systems. In Section 3, the applicability of flatness-based control to mobile robot systems is examined. In Section 4 the problem of sensor fusion for mobile robot navigation is studied. Sensor fusion of measurements coming from odometer and sonar sensors is performed for the estimation of the mobile robot's state vector. The reconstructed state vector is then used in the flatness-based control algorithm to make the robotic vehicle track the desirable trajectory. In Section 5, simulation experiments are carried out. First, it is shown how flatness-based control succeeds tracking of the desirable trajectory for the mobile robot when the complete state vector is measurable. Next, flatness-based control generates the control signal using the state vector which is reconstructed after the fusion of measurements from distributed sensors. Finally, in Section 6, concluding remarks are stated.

2. Flatness-based control

2.1 Differential flatness for finite dimensional systems

A finite dimensional system is considered. This can be written in the general form of an ODE, i.e.

$$S_i(w, \dot{w}, \ddot{w}, ..., w^i), \quad i = 1, 2, ..., q \tag{1}$$

The quantity w denotes the system variable, while w^i, $i = 1, ..., q$ are its derivatives (these and can be for instance the elements of the system's state vector). The system of Eq. (1) is said to be *differentially flat* if there exists a collection of m functions $y = (y_1, ..., y_m)$ of the system variables w_i, $i = 1, ..., s$ and of their time-derivatives, i.e.

$$y_i = \varphi(w, \dot{w}, \ddot{w}, ..., w^{\alpha_i}), \quad i = 1, ..., m \tag{2}$$

such that the following two conditions are satisfied [Roudolf, J. (2003)]:

1. There does not exist any differential relation of the form

$$R(y, \dot{y}, ..., y^{(\beta)}) = 0 \tag{3}$$

which implies that the derivatives of the flat output are not coupled in the sense of an ODE, or equivalently it can be said that the flat output is differentially independent.

2. All system variables, i.e. the components of w (elements of the system's state vectors) can be expressed using only the flat output y and its time derivatives

$$w_i = \psi_i(y, \dot{y}, ..., y^{\gamma_i}), \qquad i = i, ..., s \tag{4}$$

An equivalent definition of differentially flat systems is as follows:

Definition: The system $\dot{x} = f(x, u), \ x \in R^n, \ u \in R^m$ is differentially flat if there exist relations $h: \ R^n \times (R^m)^{r+1} \to R^m$, $\varphi: \ (R^m)^r \to R^n$ and $\psi: \ (R^m)^{r+1} \to R^m$, such that

$y = h(x,u,\dot{u},...,u^{(r)})$, $x = \varphi(y,\dot{y},...,y^{(r-1)})$ and $u = \psi(y,\dot{y},...,y^{(r-1)},y^{(r)})$. This means that all system dynamics can be expressed as a function of the flat output and its derivatives, therefore the state vector x and the control input u can be written as $x(t) = \varphi(y(t),\dot{y}(t),...,y^{(r)}(t))$ and $u(t) = \psi(y(t),\dot{y}(t),...,y^{(r+1)}(t))$.

It is noted that for linear systems the property of differential flatness is equivalent to that of controllability. Next, two examples is given to clarify the design of a differentially flat controller for finite dimensional system.

Example 1: Flatness-based control of a nonlinear system [Laroche, B. et al., (2007)]

$$\begin{aligned} \dot{x}_1 &= x_3 - x_2 u \\ \dot{x}_2 &= -x_2 + u \\ \dot{x}_3 &= x_2 - x_1 + 2x_2(u - x_2) \end{aligned} \tag{5}$$

The candidate flat differential output is $y_1 = x_1 + \frac{x_2^2}{2}$. Thus one gets:

$$\begin{aligned} y_1 &= x_1 + \frac{x_2^2}{2} \\ y_2 &= \dot{y}_1 = (x_3 - x_2 u) + x_2(u - x_2) - x_2^2 \\ y_3 &= \dot{y}_2 = \ddot{y}_1 = x_2 - x_1 + 2x_2(u - x_2) - 2x_2(u - x_2) = -x_1 + x_2 \\ \upsilon &= \dot{y}_3 = y_1^{(3)} = -x_3 + x_2 u - x_2 + u = -x_2 - x_3 + u(1 + x_2) \end{aligned} \tag{6}$$

It can be verified that property (1) holds, i.e. there does not exist any differential relation of the form $R(y,\dot{y},...,y^{(\beta)}) = 0$, and this implies that the derivatives of the flat output are not coupled. Moreover, it can be shown that property (2) also holds i.e. the components w of the system (elements of the system's state vector and control input) can be expressed using only the flat output y and its time derivatives $w_i = \psi_i(y,\dot{y},...,y^{\gamma_i})$, $i = 1,...,s$. For instance to calculate x_1 with respect to y_1, $\dot{y}_1$, $\ddot{y}_1$ and $y_1^{(3)}$ the relation of $\ddot{y}_1$ is used, i.e.:

$$x_1^2 + 2x_1(1 + \ddot{y}_1) + \ddot{y}_1^2 - 2y_1 = 0 \tag{7}$$

from which two possible solutions are derived, i.e.: $x_1 = -((1 + \ddot{y}_1) - \sqrt{1 + 2(y_1 + \ddot{y}_1)})$, $x_1 = -((1 + \ddot{y}_1) + \sqrt{1 + 2(y_1 + \ddot{y}_1)})$. Keeping the largest of these two solutions one obtains:

$$x_1 = -(1 + \ddot{y}_1 + \sqrt{1 + 2(y_1 + \ddot{y}_1)})$$
$$x_2 = \ddot{y}_1 + x_1$$
$$x_3 = \dot{y}_1 + \ddot{y}_1^2 + 2x_1\ddot{y}_1 + x_1^2 \tag{8}$$
$$u = \frac{y_1^3 + \ddot{y}_1^2 + \ddot{y}_1 + \dot{y}_1 + x_1 + 2x_1\ddot{y}_1 + x_1^2}{1 + x_1 + \ddot{y}_1}$$

Using the flat output and its derivatives, the system of Eq.(5) can be written in Brunovsky (canonical) form:

$$\frac{d}{dt}\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\upsilon \tag{9}$$

Therefore, a transformation of the system into a linear equivalent is obtained and then it is straightforward to design a a controller based on linear control theory. Thus given the reference trajectory $[x_1^r \quad x_2^r \quad x_3^r]^T$ one can find the transformed reference trajectory $[y_1^r \quad \dot{y}_1^r \quad \ddot{y}_1^r]^T$ and select the appropriate control input v that succeeds tracking. Knowing υ the control u of the initial system can be found.

Example 2: Flatness based Control of N linear coupled oscillators [Rouchon, P. (2005)]

The generalized coordinates z_i are considered and n oscillators are taken. The oscillators can be coupled through an interaction term $f_i(z_1, z_2, ..., z_N) = 0$ and through the common control input u. This means that the general oscillator model can be written as

$$\frac{d^2}{dt^2}z_i = -(\omega_i)^2 z_i + f_i(z_1, z_2, ..., z_N) + b_i u, \quad i = 1, ..., N \tag{10}$$

or, for $f_i(z_1, z_2, .., z_N) = 0$ one obtains

$$\frac{d^2}{dt^2}z_i = -(\omega_i)^2 z_i + b_i u, \quad i = 1, .., N \tag{11}$$

The terms $\omega_i > 0$ and $b_i \neq 0$ are constant parameters, while $T > 0$ and $D \neq 0$ are also defined. The objective is to find open-loop control $[0, T]$ with $t \to u(t)$ steering the system from an initial to a final state. In [Rouchon, P. (2005)] it has been shown that such control can be obtained explicitly, according to the following procedure: the Laplace transform of Eq. (11) gives

$$(s^2 + (\omega_i)^2)z_i = b_i u, \quad i = 1, .., N \tag{12}$$

Then the system can be written in the form [Lévine, J. & Nguyen, D.V. (2003)]:

$$z_i = Q_i(s)y, \;\; u = Q(s)y, \quad \text{wirh } y = \sum_{k=1}^{N} c_k z_k$$
$$Q_i(s) = \frac{b_i}{(\omega_i)^2}\prod_{k=1}^{N}(1+(\frac{s}{\omega_k})^2) \;\text{ for }\; k \neq i, \;\; Q(s) = \prod_{k=1}^{N}(1+(\frac{s}{\omega_k})^2), \;\; c_k = \frac{1}{Q_k(j\omega_k)} \;\in R \tag{13}$$

The real coefficients q_k^i and q_k are defined as follows [Rouchon, P. (2005)]: $Q_i(s) = \sum_{k=0}^{N-1} q_k^i s^{2k}$ and $Q(s) = \sum_{k=0}^{N} q_k s^{2k}$. This enables to express both the system parameters $x(t)$ and the control input $u(t)$ as functions of the flat output $y(t)$, i.e.

$$x_i(t) = \sum_{k=0}^{n-1} q_k^i y^{(2k)}(t), \quad \upsilon(t) = \sum_{k=0}^{n} q_k y^{(2k)}(t) \tag{14}$$

which satisfy Eq. (11), with $z_i = x_i$ and $u = v$. Moreover it holds that

$$y(t) = \sum_k c_k x_k(t) \tag{15}$$

There is a one to one linear correspondence between the trajectories of Eq. (11) and the arbitrary smooth time function y. More precisely, any piecewise continuous open-loop control $u(t), \;\; t \in [0,T]$ steering from the steady-state $z_i(0) = 0$ to the steady-state $z_i(T) = \frac{b_i}{(\omega_i)^2} D$ can be written as

$$u(t) = \sum_{k=0}^{N} q_k y^{(2k)}(t) \tag{16}$$

for all functions $y(t)$ such that $y(0) = 0$, $y(T) = D \;\; \forall \;\; i \;\; \{1,2,..,2n-1\}$, $y^{(i)}(0) = y^{(i)}(T) = 0$.

The results can be extended to the case of a harmonic oscillator with damping. In that case Eq. (11) is replaced by [Rouchon, P. (2005)]

$$\frac{d^2 z_i}{dt^2} = -\omega_i^2 z_i - 2\xi_i \omega_i \dot{z}_i + b_i u, \;\; i = 1,...,n \tag{17}$$

where the damping coefficient is $\xi_i \geq 0$. In that case one obtains

$$z_i = Q_i(s)y, \;\; u = Q(s)y \tag{18}$$

with

$$Q_i(s) = \frac{b_i}{(\omega_i)^2}\prod_{k=1}^{n}(1+2\xi_k(\frac{s}{\omega_k})+(\frac{s}{\omega_k})^2) \;\; k \neq i, \quad Q(s) = \prod_{k=1}^{n}(1+2\xi_k(\frac{s}{\omega_k})+(\frac{s}{\omega_k})^2) \tag{19}$$

which proves again that the system's parameters (state variables) and the control input can be written as functions of the flat output y and its derivatives. In that case the flat output is of the form:

$$y = \sum_{k=1}^{n} c_k z_k + d_k s z_k \tag{20}$$

where $s = d/dt$ and the coefficients c_k and d_k can be computed explicitly. According to [Lévine, J. & Nguyen, D.V. (2003)] explicit descriptions of the system parameters via an arbitrary function y (flat output) and its derivatives are possible for any controllable linear system of finite dimension (controllability is equivalent to flatness).

2.2 Flatness-based control for the unicycle robot

Several mobile robots with non-holonomic motion constraints, are differentially flat systems. The conditions for motion without sliding are given in the sequel:

$$\begin{aligned} \dot{x} &= v\cos\theta \\ \dot{y} &= v\sin\theta \\ \dot{\theta} &= \frac{v}{l}\tan\varphi \end{aligned} \tag{21}$$

where v is the velocity of the vehicle, l is the vehicle's length, θ is the angle between the transversal axis of the vehicle and axis OX, and ϕ is the angle of the steering wheels with respect to the transversal axis of the vehicle. The flat output is the cartesian position of the vehicle's center of gravity, denoted as $\eta = (x, y)$, while the other model parameters can be written as:

$$v = \pm \| \dot{\eta} \|, \quad \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} = \frac{\dot{\eta}}{v}, \quad \tan(\varphi) = l \det(\dot{\eta}\, \ddot{\eta}) / v^3 \tag{22}$$

These formulas show simply that is the tangent angle of the curve traced by P and $\tan(\phi)$ is the associated curvature.
With reference to a generic driftless nonlinear system

$$\dot{q} = G(q)w, \quad q \in R^n, \quad w \in R^m \tag{23}$$

the dynamic feedback linearization consists in finding a feedback compensator of the form

$$\begin{aligned} \dot{\xi} &= \alpha(q,\xi) + b(q,\xi)u \\ w &= c(q,\xi) + d(q,\xi)u \end{aligned} \tag{24}$$

with state $\xi \in R^{\nu}$ and input $u \in R^m$, such that the closed-loop system of Eq. (23) and Eq. (24) is equivalent under a state transformation $z = T(q,\xi)$ to a linear system. The starting point

is the definition of an m-dimensional output $\eta = h(q)$ to which a desired behavior can be assigned. One then proceeds by successively differentiating the output until the input appears in a non-singular way. If the sum of the output differentiation orders equals the dimension $n+v$ of the extended state space, full input-state-output linearization is obtained (In this case η is also called a flat output). The closed-loop system is then equivalent to a set of decoupled input-output chains of integrators from u_i to η_i $(i = 1, 2, .., m)$. The exact linearization procedure is illustrated for the unicycle model of Eq. (21). As flat output the coordinates of the center of gravity of the vehicle is considered $\eta = (x, y)$. Differentiation with respect to time then yields [Oriolo, G. et al.,(2002)]

$$\dot{\eta} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{pmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \end{pmatrix} \cdot \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{25}$$

showing that only v affects $\dot{\eta}$, while the angular velocity ω cannot be recovered from this first-order differential information. To proceed, one needs to add an integrator (whose state is denoted by ξ) on the linear velocity input

$$v = \xi, \quad \dot{\xi} = \alpha \Rightarrow \dot{\eta} = \xi \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} \tag{26}$$

where α denotes the linear acceleration of the robotic vehicle. Differentiating further one obtains

$$\ddot{\eta} = \dot{\xi} \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} + \xi \dot{\theta} \begin{pmatrix} \sin(\theta) \\ \cos(\theta) \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\xi \sin(\theta) \\ \sin(\theta) & \xi \cos(\theta) \end{pmatrix} \begin{pmatrix} \alpha \\ \omega \end{pmatrix} \tag{27}$$

and the matrix multiplying the modified input (a, ω) is nonsingular if $\xi \neq 0$. Under this assumption one defines

$$\begin{pmatrix} \alpha \\ \omega \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\xi \sin(\theta) \\ \sin(\theta) & \xi \cos(\theta) \end{pmatrix}^{-1} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \tag{28}$$

$\ddot{\eta}$ is denoted as

$$\ddot{\eta} = \begin{pmatrix} \ddot{\eta}_1 \\ \ddot{\eta}_2 \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = u \tag{29}$$

which means that the desirable linear acceleration and the desirable angular velocity can be expressed using the transformed control inputs u_1 and u_2. Then, the resulting dynamic compensator is (return to the initial control inputs v and ω)

$$\begin{aligned} \dot{\xi} &= u_1 \cos(\theta) + u_2 \sin(\theta) \\ v &= \xi \\ \omega &= \frac{u_2 \cos(\theta) - u_1 \sin(\theta)}{\xi} \end{aligned} \tag{30}$$

Being $\xi \in R$, it is $n + v = 3 + 1 = 4$, equal to the output differentiation order in Eq. (29). In the new coordinates

$$\begin{aligned} z_1 &= x \\ z_2 &= y \\ z_3 &= \dot{x} = \xi \cos(\theta) \\ z_4 &= \dot{y} = \xi \sin(\theta) \end{aligned} \tag{31}$$

The extended system is thus fully linearized and described by the chains of integrators, in Eq. (29), and can be rewritten as

$$\begin{aligned} \ddot{z}_1 &= u_1 \\ \ddot{z}_2 &= u_2 \end{aligned} \tag{32}$$

The dynamic compensator of Eq. (30) has a potential singularity at $\xi = v = 0$, i.e. when the unicycle is not rolling. The occurrence of such singularity is structural for non-holonomic systems. This difficulty must be obviously taken into account when designing control laws on the equivalent linear model.

A nonlinear controller for output trajectory tracking, based on dynamic feedback linearization, is easily derived. Assume that the robot must follow a smooth trajectory $(x_d(t), y_d(t))$ which is persistent, i.e. for which the nominal velocity $v_d = (\dot{x}_d^2 + \dot{y}_d^2)^{1/2}$ along the trajectory never goes to zeros (and thus singularities are avoided). On the equivalent and decoupled system of Eq. (32), one can easily design an exponentially stabilizing feedback for the desired trajectory, which has the form

$$\begin{aligned} u_1 &= \ddot{x}_d + k_{p_1}(x_d - x) + k_{d_1}(\dot{x}_d - \dot{x}) \\ u_2 &= \ddot{y}_d + k_{p_1}(y_d - y) + k_{d_1}(\dot{y}_d - \dot{y}) \end{aligned} \tag{33}$$

which results in the following error dynamics for the closed-loop system

$$\begin{aligned} \ddot{e}_x + k_{d_1}\dot{e}_x + k_{p_1}e_x &= 0 \\ \ddot{e}_y + k_{d_2}\dot{e}_y + k_{p_2}e_y &= 0 \end{aligned} \tag{34}$$

where $e_x = x - x_d$ and $e_y = y - y_d$. The proportional-derivative (PD) gains are chosen as $k_{p_i} > 0$ and $k_{d_i} > 0$ for $i = 1, 2$. Knowing the control inputs u_1, u_2 for the linearized system

one can calculate the control inputs v and ω applied to the robotic vehicle, using Eq. (24). The above result is valid, provided that the dynamic feedback compensator does not need the singularity $v = \xi_i = 0$. The following theorem assures the avoidance of singularities in the proposed control law [Oriolo, G. et al., (2002)]:

Theorem: Let λ_{11}, λ_{12} and λ_{21}, λ_{22}, be respectively the eigenvalues of two equations of the error dynamics, given in Eq. (34). Assume that, for $i = 1,2$ it is $\lambda_{i1} < \lambda_{i2} < 0$ (negative real eigenvalues), and that λ_{i2} is sufficiently small. If

$$\min_{t \geq 0} \| \begin{bmatrix} \dot{x}_d(t) \\ \dot{y}_d(t) \end{bmatrix} \| > | \begin{bmatrix} \dot{\varepsilon}_x^0 \\ \dot{\varepsilon}_y^0 \end{bmatrix} | \tag{35}$$

with $\dot{\varepsilon}_x^0 = \dot{\varepsilon}_x(0) \neq 0$ and $\dot{\varepsilon}_y^0 = \dot{\varepsilon}_y(0) \neq 0$, then the singularity $\xi = 0$ is never met.

3. Fusion of distributed measurements using the Extended Kalman Filter

The control law for the unicycle robot described in subsection 2.2. was based on the assumption that the vehicle's state vector $[x(k), y(k), \theta(k)]$ was measurable at every time instant. Here, the case in which the vehicle's state vector is reconstructed through the fusion of measurements received from distributed sensors (such as odometer or sonar sensors) will be examined.

The first approach to be analyzed is that of fusion of measurements coming from distributed sensors, with the use of nonlinear filtering methods such as Extended Kalman Filtering (EKF). The fused data are used to reconstruct the state vector of a mobile robot, and the estimated state vector is in turn used in a control-loop.

Extended Kalman Filtering for the nonlinear state-measurement model is revised. The following nonlinear time-invariant state model is now considered [Rigatos, G.G. & Tzafestas, S.G. (2007)]:

$$\begin{aligned} x(k+1) &= \phi(x(k)) + L(k)u(k) + w(k) \\ z(k) &= \gamma(x(k)) + v(k) \end{aligned} \tag{36}$$

where $w(k)$ and $v(k)$ are uncorrelated, Gaussian zero-mean noise processes with covariance matrices $Q(k)$ and $R(k)$ respectively. The operators $\phi(x)$ and $\gamma(x)$ are given by, $\phi(x) = [\phi_1(x), \phi_2(x), ..., \phi_m(x),]^T$, and $\gamma(x) = [\gamma_1(x), \gamma_2(x), ..., \gamma_p(x)]^T$, respectively. It is assumed that ϕ and γ are sufficiently smooth in x so that each one has a valid series Taylor expansion. Following a linearization procedure, ϕ is expanded into Taylor series about $\hat{x}$:

$$\phi(x(k)) = \phi(\hat{x}(k)) + J_\phi(\hat{x}(k))[x(k) - \hat{x}(k)] + ... \tag{37}$$

where $J_\phi(x)$ is the Jacobian of ϕ calculated at $\hat{x}(k)$:

$$J_\phi(x) = \frac{\partial \phi}{\partial x}|_{x=\hat{x}(k)} = \begin{pmatrix} \frac{\partial \phi_1}{\partial x_1} & \frac{\partial \phi_1}{\partial x_2} & \cdots & \frac{\partial \phi_1}{\partial x_N} \\ \frac{\partial \phi_2}{\partial x_1} & \frac{\partial \phi_2}{\partial x_2} & \cdots & \frac{\partial \phi_2}{\partial x_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial \phi_m}{\partial x_1} & \frac{\partial \phi_m}{\partial x_2} & \cdots & \frac{\partial \phi_m}{\partial x_N} \end{pmatrix} \tag{38}$$

Likewise, γ is expanded about $\hat{x}^-(k)$

$$\gamma(x(k)) = \gamma(\hat{x}(k)) + J_\lambda(\hat{x}(k))[x(k) - \hat{x}(k)] + \ldots \tag{39}$$

where $\hat{x}^-(k)$ is the prior to time instant k estimation of the state vector $x(k)$, and $\hat{x}(k)$ is the estimation of $x(k)$ at time instant k. The Jacobian $J_\gamma(x)$ is

$$J_\gamma(x) = \frac{\partial \gamma}{\vartheta x}|_{x=\hat{x}^-(k)} = \begin{pmatrix} \frac{\partial \gamma_1}{\partial x_1} & \frac{\partial \gamma_1}{\partial x_2} & \cdots & \frac{\partial \gamma_1}{\partial x_N} \\ \frac{\partial \gamma_2}{\partial x_1} & \frac{\partial \gamma_2}{\partial x_2} & \cdots & \frac{\partial \gamma_2}{\partial x_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial \gamma_p}{\partial x_1} & \frac{\partial \gamma_p}{\partial x_2} & \cdots & \frac{\partial \gamma_p}{\partial x_N} \end{pmatrix} \tag{40}$$

The resulting expressions create first order approximations of ϕ and γ. Thus the linearized version of the plant is obtained:

$$x(k+1) = \varphi(\hat{x}(k)) + J_\varphi(\hat{x}(k))[x(k) - \hat{x}(k)] + w(k)$$
$$z(k) = \gamma(\hat{x}^-(k)) + J_\gamma(\hat{x}^-(k))[x(k) - \hat{x}^-(k)] + v(k)$$

Now, the EKF recursion is as follows: First the time update is considered: by $\hat{x}(k)$ the estimation of the state vector at instant k is denoted. Given initial conditions $\hat{x}^-(0)$ and $P^-(0)$the recursion proceeds as:

- *Measurement update*. Acquire $z(k)$ and compute:

$$K(k) = P^-(k)J_\gamma^T(\hat{x}^-(k))[J_\gamma(\hat{x}^-(k))P^-(k)J_\gamma^T(\hat{x}^-(k)) + R(k)]^{-1}$$
$$\hat{x}(k) = \hat{x}^-(k) + K(k)[z(k) - \gamma(\hat{x}^-(k))] \tag{41}$$
$$P(k) = P^-(k) - K(k)J_\gamma(\hat{x}^-(k))P^-(k)$$

- *Time update*. Compute:

$$P^{-}(k+1) = J_{\phi}(\hat{x}(k))P(k)J_{\phi}^{T}(\hat{x}(k)) + Q(k)$$
$$\hat{x}^{-}(k+1) = \phi(\hat{x}(k)) + L(k)u(k) \tag{42}$$

The schematic diagram of the EKF loop is given in Fig. 1(a).

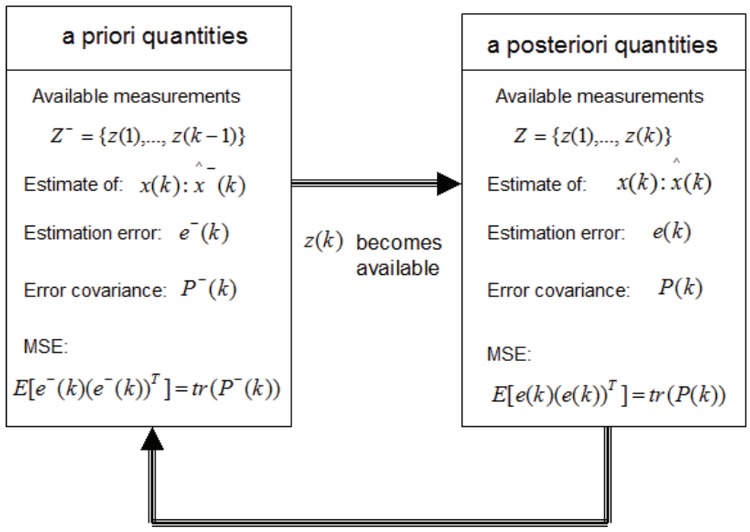

Fig. 1. (a) Schematic diagram of the Extended Kalman Filter Loop

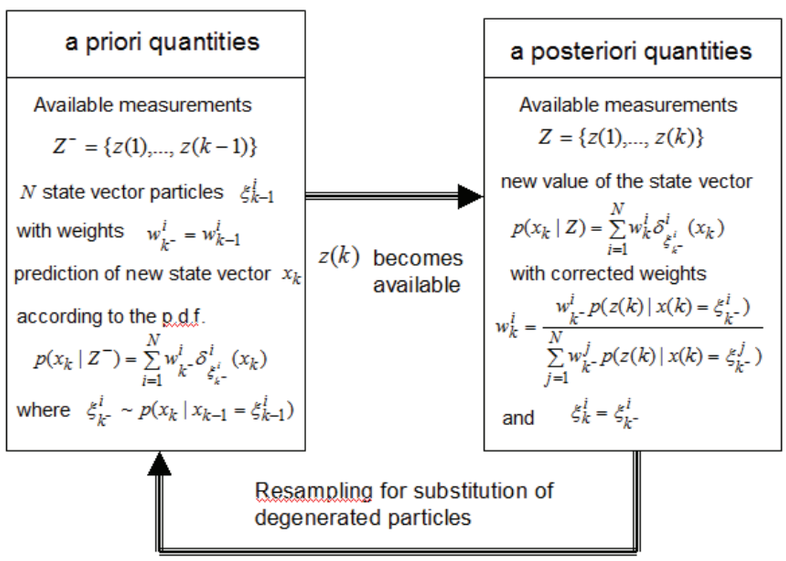

Fig. 1. (b) Schematic diagram of the Particle Filter loop

4. Particle Filtering for the nonlinear state-measurement model

4.1 Particle Filtering with sequential importance resampling

Next, the problem of fusing measurements coming from distributed sensors will be solved using the Particle Filtering method. The fused data are used again to estimate the state vector of a mobile robot and the reconstructed state vector will be used in closed-loop control.

In the general case the equations of the optimal filter used for the calculation of the state-vector of a nonlinear dynamical system do not have an explicit solution. This happens for instance when the process noise and the noise of the output measurement do not follow a Gaussian distribution. In that case, approximation through Monte-Carlo methods can used. As in the case of the Kalman Filter or the Extended Kalman Filter the particles filter consists of the measurement update (correction stage) and the time update (prediction stage) [Thrun, S. et al., (2005)].

a. *The prediction stage*:

The prediction stage calculates $p(x(k) \mid Z^-)$ where $Z^- = \{z(1),..,z(n-1)\}$, using:

$$p(x(k-1) \mid Z^-) = \sum_{i=1}^{N} w_{k-1}^i \delta_{\xi_{k-1}^i}(x(k-1)) \tag{43}$$

while from Bayes formula it holds $p(x(k-1) \mid Z^-) = \int p(x(k) \mid x(k-1)) p(x(k-1) \mid Z^-) dx$. This finally gives

$$p(x(k) \mid Z^-) = \sum_{i=1}^{N} w_{k-1}^i \delta_{\xi_{k^-}^i}(x(k))$$

$$\text{with } \xi_{k^-}^i \sim p(x(k) \mid x(k-1)) = \xi_{k-1}^i \tag{44}$$

The meaning of Eq. (44) is as follows: the state equation of the nonlinear system of Eq. (36) is executed N times, starting from the N previous values of the state vectors $x(k-1) = \xi_{k-1}^i$ with the use of Eq. (36). This means that the value of the state vector which is calculated in the prediction stage is the result of the weighted averaging of the state vectors which were calculated after running the state equation, starting from the N previous values of the state vectors ξ_{k-1}^i.

b. The correction stage

The a-posteriori probability density was performed using Eq. (44). Now a new position measurement $z(k)$ is obtained and the objective is to calculate the corrected probability density $p(x(k) \mid Z)$, where $Z = \{z(1), z(2),.., z(k)\}$. From Bayes law it holds that $p(x(k) \mid Z) = \frac{p(Z \mid x(k)) p(x(k))}{p(Z)}$, which finally results into

$$p(x(k) \mid Z^-) = \sum_{i=1}^{N} w_{k-1}^i \delta_{\xi_{k^-}^i}(x(k))$$

$$\text{where} \quad w_k^i = \frac{w_{k^-}^i \, p(z(k) \mid x(k) = \xi_{k^-}^i)}{\sum_{j=1}^{N} w_{k^-}^j \, p(z(k) \mid x(k) = \xi_{k^-}^j)} \tag{45}$$

Eq. (45) denotes the corrected value for the state vector. The recursion of the Particle Filter proceeds in a way similar to the update of the Kalman Filter or the Extended Kalman Filter, i.e.:

Measurement update: Acquire $z(k)$ and compute the new value of the state vector

$$p(x(k) \mid Z) = \sum_{i=1}^{N} w_k^i \delta_{k^-}^i (x(k))$$

$$\text{with corrected weights} \quad w_k^i = \frac{w_{k^-}^i \, p(z(k) \mid x(k) = \xi_{k^-}^i)}{\sum_{j=1}^{N} w_{k^-}^i \, p(z(k) \mid x(k) = \xi_{k^-}^i)^i} \quad \text{and} \quad \xi_k^i = \xi_{k^-}^i \tag{46}$$

Resample for substitution of the degenerated particles.

Time update: compute state vector $x(k+1)$ according to

$$p(x(k+1) \mid Z) = \sum_{i=1}^{N} w_k^i \delta_{\xi_k^i} (x(k)) \tag{47}$$

$$\text{where} \quad \xi_k^i \sim p(x(k+1) \mid x(k) = \xi_k^i)$$

The stages of state vector estimation with the use of the particle filtering algorithm are depicted in Fig. 1(b).

4.2 Resampling issues in particle filtering

a. Degeneration of particles

The algorithm of particle filtering which is described through Eq. (44) and Eq. (45) has a significant drawback: after a certain number of iterations k, almost all the weights w_k^i become 0. In the ideal case all the weights should converge to the value $1/N$, i.e. the particles should have the same significance. The criterion used to define a sufficient number of particles is $N_k^{eff} = 1/\sum_{i=1}^{N} w_k^{i\,2} \in [1, N]$. When N_k^{eff} is close to value N then all particles have almost the same significance. However using the algorithm of Eq. (44) and Eq. (45) results in $N_k^{eff} \to 1$, which means that the particles are degenerated, i.e. they lose their effectiveness. Therefore, it is necessary to modify the algorithm so as to assure that degeneration of the particles will not take place [Crisan, D. & Doucet, A. (2002)].

When N_k^{eff} is small then most of the particles have weights close to 0 and consequently they have a negligible contribution to the estimation of the state vector. The concept proposed to overcome this drawback of the algorithm is to weaken this particle in favor of particles that have a non-negligible contribution. Therefore, the particles of low weight factors are

removed and their place is occupied by duplicates of the particles with high weight factors. The total number of particles remains unchanged (equal to N) and therefore this procedure can be viewed as a "resampling" or "redistribution" of the particles set.

The particles resampling mentioned above maybe slow if not appropriately tuned. There are improved versions of it which substitute the particles of low importance with those of higher importance [Arulampalam, S. et al., (2002)], [Kitagawa, (1996)]. A first choice would be to perform a multinomial resampling. N particles are chosen between $\{\xi_k^1,...,\xi_k^N\}$ and the corresponding weights are $\{w_k^1,...,w_k^N\}$. The number of times each particle is selected is given by $[j_1,..,j_N]$. Thus a set of N particles is again created, the elements of which are chosen after sampling with the discrete distribution $\sum_{i=1}^{N} w_k^i \delta_{\xi_k^i}(x)$. The particles $\{\xi_k^1,...,\xi_k^N\}$ are chosen according to the probabilities $\{w_k^1,..,w_k^N\}$. The selected particles are assigned with equal weights $1/N$.

b. Other approaches to the implementation of resampling

Although sorting of the particles' weights is not necessary for the convergence of the particle filter algorithm, there are variants of the resampling procedure of $(\xi_k^i, w_k^i,\ \ i=1,..,N)$ which are based on previous sorting in decreasing order of the particles' weights. It is noted that efficient sorting approaches make the complexity of the particle filtering to be $O(N\log(N))$, while the avoidance of resampling could result in a faster algorithm of complexity $O(N)$. Sorting of particles' weights gives $w^{s[1]} > w^{s[2]} > ... > w^{s[N]}$. A random numbers generator is evoked and the resulting numbers $u^{i:N} \sim U[0,1]$ fall in the partitions of the interval $[0,1]$. The width of these partitions is w^j and thus a redistribution of the particles is generated. For instance, in a wide partition of width w^j will be assigned more particles than to a narrow partition of width w^m.

Two other methods that have been proposed for the implementation of resampling in Particle Filtering are explained in the sequel. These are Kitagawa's approach and the residuals resampling approach [Kitagawa, G. (1996)]. In Kitagawa's resampling the speed of the resampling procedure is increased by using less the random numbers generator. The weights are sorted again in decreasing order $w^{s[j]}$ so as to cover the region that corresponds to the interval [0,1]. Then the random numbers generator is used to produce the variable u_i according to according to $u_1 \sim U[0,1/N]$, and $u^i = u^1 + \frac{i}{N},\ i=2,..,N$. The rest of the variables u^i are produced in a deterministic way [Campillo, F. (2006)].

In the residuals resampling approach, the redistribution of the residuals is performed as follows: at a first stage particle ξ^i is chosen in a deterministic way $[w^i/N]$ times (with rounding). The residual weights are $\tilde{w}^i = w^i - N[w^i/N]$ and are normalized. Thus, a probability distribution is generated. The rest of the particles are selected according to multinomial resampling . The method can be applied if the number $\tilde{N}$ which remains at the second stage is small, i.e. when $N^{eff} = 1/\sum_{i=1}^{N} w_i^2$ is small.

5. Simulation tests

5.1 Flatness-based control using a state vector estimated by EKF

The application of EKF to the fusion of data that come from different sensors is examined first [Rigatos, G.G. & Tzafestas, S.G. (2007)]. A unicycle robot is considered. Its continuous-time kinematic equation is:

$$\dot{x}(t) = v(t)\cos(\theta(t)), \quad \dot{y}(t) = v(t)\sin(\theta(t)), \quad \dot{\theta}(t) = \omega(t) \tag{48}$$

Encoders are placed on the driving wheels and provide a measure of the incremental angles over a sampling period T. These odometric sensors are used to obtain an estimation of the displacement and the angular velocity of the vehicle $v(t)$ and $\omega(t)$, respectively. These encoders introduce incremental errors, which result in an erroneous estimation of the orientation θ. To improve the accuracy of the vehicle's localization, measurements from sonars can be used. The distance measure of sonar i from a neighboring surface P_j is thus taken into account (see Fig. 2(a) and Fig. 2(b)). Sonar measurements may be affected by white Gaussian noise and also by crosstalk interferences and multiples echoes.

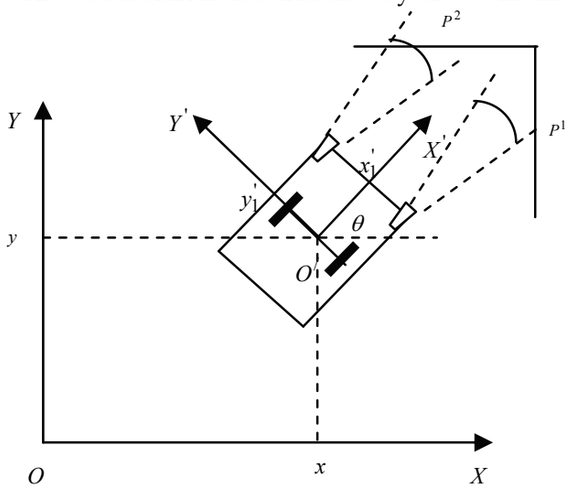

Fig. 2. (a). Mobile robot with odometric and sonar sensors

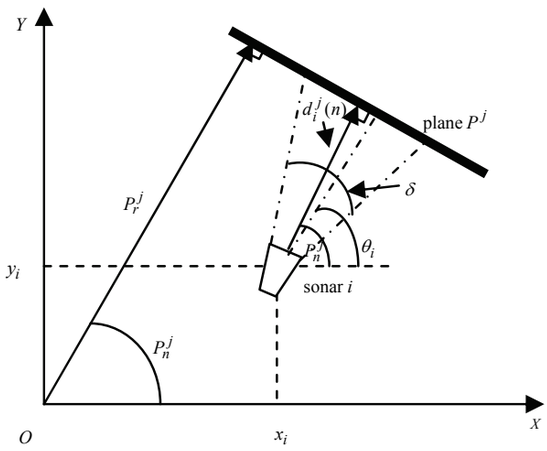

Fig. 2. (b). Orientation of the sonar i

The inertial coordinates system OXY is defined. Furthermore the coordinates system $O'X'Y'$ is considered (Fig. 2(a)). $O'X'Y'$ results from OXY if it is rotated by an angle θ. The coordinates of the center of the wheels axis with respect to OXY are (x, y), while the coordinates of the sonar i that is mounted on the vehicle, with respect to $O'X'Y'$ are x_i', y_i'. The orientation of the sonar with respect to $O'X'Y'$ is θ_i'. Thus the coordinates of the sonar with respect to OXY are (x_i, y_i) and its orientation is θ_i, and are given by:

$$\begin{aligned} x_i(k) &= x(k) + x_i' \sin(\theta(k)) + y_i' \sin(\theta(k)) \\ x_i(k) &= x(k) + x_i' \sin(\theta(k)) + y_i' \sin(\theta(k)) \\ \theta_i(k) &= \theta(k) + \theta_i \end{aligned} \tag{49}$$

Each plane P^j in the robot's environment can be represented by P_r^j and P_n^j (Fig. 2(b)), where (i) P_r^j is the normal distance of the plane from the origin O, (ii) P_n^j is the angle between the normal line to the plane and the x-direction.

The sonar i is at position $(x_i(k), y_i(k))$ with respect to the inertial coordinates system OXY and its orientation is $\theta_i(k)$. Using the above notation, the distance of the sonar i, from the plane P^j is represented by P_r^j, P_n^j (see Fig. 2(b)):

$$d_i^k(k) = P_r^j - x_i(k)\cos(P_n^j) - y_i(k)\sin(P_n^j) \tag{50}$$

where $P_n^j \in [\theta_i(n) - \delta/2, \theta_i(n) + \delta/2]$, and δ is the width of the sonar beam. Assuming a constant sampling period $\Delta t_k = T$ the measurement equation is $z(k+1) = \gamma(x(k)) + v(k)$, where $z(k)$ is the vector containing sonar and odometer measures and $v(k)$ is a white noise sequence $\sim N(0, R(kT))$. The dimension p_k of $z(k)$ depends on the number of sonar sensors. The measure vector $z(k)$ can be decomposed in two sub-vectors

$$\begin{aligned} z_1(k+1) &= [x(k) + v_1(k), y(k) + v_2(k), \theta(k) + v_3(k)] \\ z_2(k+1) &= [d_1^j(k) + v_4(k), ..., d_{n_s}^j(k) + v_{3+n_2}(k)] \end{aligned} \tag{51}$$

with $i = 1, 2, ..., n_s$, where n_s is the number of sonars, $d_i^j(k)$ is the distance measure with respect to the plane P^j provided by the i-th sonar and $j = 1, .., n_p$ where n_p is the number of surfaces. By definition of the measurement vector one has that the output function $\gamma(x(k))$ is given by $\gamma(x(k)) = [x(k), y(k), \theta(k), d_1^1(k), d_2^2(k), ..., d_{n_s}^{n_p}]^T$. The robot state is $[x(k), y(k), \theta(k)]^T$ and the control input is denoted by $U(k) = [u(k), \omega(k)]^T$.

In the simulation tests, the number of sonar is taken to be $n_s = 1$, and the number of planes $n_p = 1$, thus the measurement vector becomes s. To obtain the Extended Kalman Filter (EKF), the kinematic model of the vehicle is linearized about the estimates $\hat{x}(k)$ and $\hat{x}^-(k)$, the control input $U(k-1)$ is applied.

The *measurement update* of the EKF is

$$K(k) = P^-(k) J_\gamma^T(\hat{x}^-(k))[J_\gamma(\hat{x}^-(k)) P^-(k) J_\gamma^T(\hat{x}^-(k)) + R(k)]^{-1}$$

$$\hat{x}(k) = \hat{x}^-(k) + K(k)[z(k) - \gamma(\hat{x}^-(k))]$$

$$P(k) = P^-(k) - K(k) J_\gamma(\hat{x}^-(k)) P^-(k)$$

The *time update* of the EKF is

$$P^-(k+1) = J_\phi(\hat{x}(k)) P(k) J_\phi^T(\hat{x}(k)) + Q(k)$$

$$\hat{x}^-(k+1) = \phi(\hat{x}(k)) + L(k)u(k)$$

where $L(n)=\begin{pmatrix} T\cos(\theta(k)) & 0 \\ T\sin(\theta(k)) & 0 \\ 0 & T \end{pmatrix}$ and $J_{\varphi}(\hat{x}(k))=\begin{pmatrix} 1 & 0 & -v(k)\sin(\theta)T \\ 0 & 1 & -v(k)\cos(\theta)T \\ 0 & 0 & 1 \end{pmatrix}$

while $Q(k)=diag[\sigma^2(k),\sigma^2(k),\sigma^2(k)]$, with $\sigma^2(k)$ chosen to be 10^{-3}, and

$\phi(\hat{x}(k))=[\hat{x}(k),\hat{y}(k),\hat{\theta}(k)]^T$, $\gamma(\hat{x}(k))=[\hat{x}(k),\hat{y}(k),\hat{\theta}(k),d(k)]^T$, i.e.

$$\gamma(\hat{x}(k))=\begin{pmatrix} \hat{x}(k) \\ \hat{y}(k) \\ \hat{\theta}(k) \\ P_r^j-x_i(k)\cos(P_n^j)-y_i(k)\sin(P_n^j) \end{pmatrix} \tag{52}$$

Assuming one sonar $n_s=1$, and one plane P^1, $n_p=1$ in the mobile robot's neighborhood one gets

$$J_{\gamma}^{T}(\hat{x}^{-}(k))=\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -\cos(P_n^j) & -\sin(P_n^j) & \{x_i^{'}\cos(\theta-P_n^j)-y_i^{'}\sin(\theta-P_n^j)\} \end{pmatrix} \tag{53}$$

The vehicle is steered by a dynamic feedback linearization control algorithm which is based on flatness-based control [Oriolo, G. et al., (2002)]:

$$\begin{aligned} u_1&=\ddot{x}_d+K_{p_1}(x_d-x)+K_{d_1}(\dot{x}_d-\dot{x}) \\ u_2&=\ddot{y}_d+K_{p_2}(y_d-y)+K_{d_2}(\dot{y}_d-\dot{y}) \\ \dot{\xi}&=u_1\cos(\theta)+u_2\sin(\theta) \\ v&=\xi,\quad \omega=\frac{u_2\cos(\theta)-u_1\sin(\theta)}{\xi} \end{aligned} \tag{54}$$

The following initialization is assumed (see Fig. 3(a)): (i) vehicle's initial position in OXY: $x(0)=0\,m$, $y(0)=0\,m$, $\theta(0)=45.0^o$, (ii) position of the sonar in $O'X'Y'$: $x_1^{'}=0.5\,m$, $y_1^{'}=0.5\,m$, $\theta_1^{'}=0^o$, (iii) position of the plane P^1: $P_r^1=15.5\,m$, $P_n^1=45^o$, (iv) state noise $w(k)=0$, $\hat{P}(0)=diag[0.1,0.1,0.1]$ and $R=diag[10^{-3},10^{-3},10^{-3}]$, (v) Kalman Gain $K(k)\in R^{3\times 4}$

The use of EKF for fusing the data that come from odometric and sonar sensors provides an estimation of the state vector $[x(t),y(t),\theta(t)]$ and enables the successful application of

nonlinear steering control of Eq. (54). For the case of motion along a straight line on the 2D-plane, the obtained results are depicted in Fig. 3(a). Moreover, results on the tracking of a circular reference path are given in Fig. 4(a), while the case of tracking of an eight-shaped reference path is depicted in Fig. 5(a). Tracking experiments for EKF-based state estimation were completed in the case of a curved path as the one shown in Fig. 6(a).

5.2 Flatness-based control using a state vector estimated by Particle Filtering

The particle filter can also provide solution to the sensor fusion problem. The mobile robot model described in Eq. (48), and the control law given in Eq. (54) are used again. The number of particles was set to $N = 1000$.

The *measurement update* of the PF is $p(x(k) \mid Z) = \sum_{i=1}^{N} w_k^i \delta_{\xi_{k^-}^i}(x(k))$ with $w_k^i = \frac{w_{k^-}^i \, p(z(k) \mid x(k) = \xi_{k^-}^i)}{\sum_{j=1}^{N} w_k^j \, p(z(k) \mid x(k) = \xi_{k^-}^j)}$ where the measurement equation is given by $\hat{z}(k) = z(k) + v(k)$ with $z(k) = [x(k), y(k), \theta(k), d(k)]^T$, and $v(k) =$ measurement noise.

The *time update* of the PF is $p(x(k+1) \mid Z) = \sum_{i=1}^{N} w_k^i \delta_{\xi_k^i}(x(k))$ where $\xi_k^i \sim p(x(k+1) \mid x(k) = \xi_k^i)$ and the state equation is $\hat{x}^- = \varphi(x(k)) + L(k)U(k)$, where $\phi(x(k))$, $L(k)$ and $U(k)$ are defined in subsection 5.1. At each run of the time update of the PF, the state vector estimation $\hat{x}^-(k+1)$ is calculated N times, starting each time from a different value of the state vector ξ_k^i. The measurement noise distribution was assumed to be Gaussian. As the number of particles increases, the performance of the particle filter-based tracking algorithm also improves, but this happens at higher demand for computational resources. Control of the diversity of the particles through the tuning of the resampling procedure may also affect the performance of the algorithm. The obtained results are given in Fig. 3(b) for the case of motion along a straight line on the 2D plane. Additionally, results on the tracking of a circular reference path are given in Fig. 4(b), while the case of tracking of an eight-shaped reference path is depicted in Fig. 5(b). Tracking experiments for PF-based state estimation were completed in the case of a curved path as the one shown in Fig. 6(b).

From the depicted simulation experiments it can be deduced that the particle filter for a sufficiently large number of particles can have good performance, in the problem of estimation of the state vector of the mobile robot, without being subject to the constraint of Gaussian distribution for the obtained measurements. The number of particles influences the performance of the particle filter algorithm. The accuracy of the estimation succeeded by the PF algorithm improves as the number of particles increases. The initialization of the particles, (state vector estimates) may also affect the convergence of the PF towards the real value of the state vector of the monitored system. It should be also noted that the calculation time is a critical parameter for the suitability of the PF algorithm for real-time applications.

When it is necessary to use more particles, improved hardware and parallel processing available to embedded systems, enable the PF to be implemented in real-time systems [Yang, N. et al., (2005)].

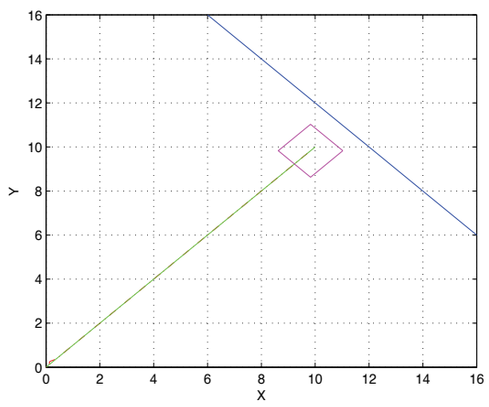

Fig. 5. (a) Desirable trajectory (continuous line) and trajectory using EKF fusion based on odometric and sonar measurements, when tracking a straight line.

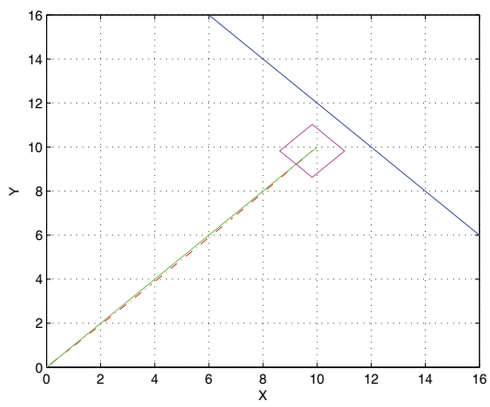

Fig. 5. (b) Desirable trajectory (continuous line) and trajectory using PF fusion based on odometric and sonar measurements, when tracking a straight line.

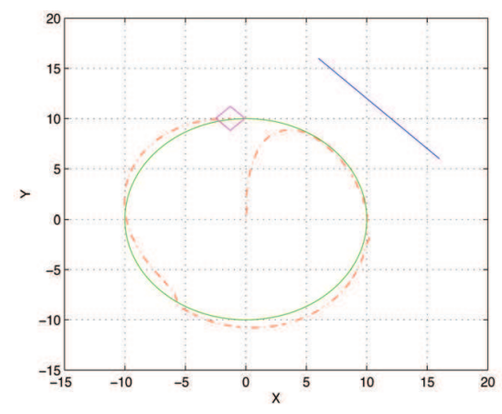

Fig. 4. (a) The trajectory of the mobile robot (dashed line) tracks the reference circular path (continuous line) when the robot's state vector is estimated with the use of Extended Kalman Filtering.

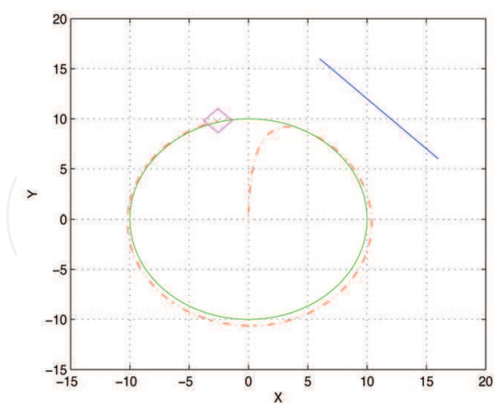

Fig. 4. (b) The trajectory of the mobile robot (dashed line) tracks the reference circular path (continuous line) when the robot's state vector is estimated with the use of particle filtering.

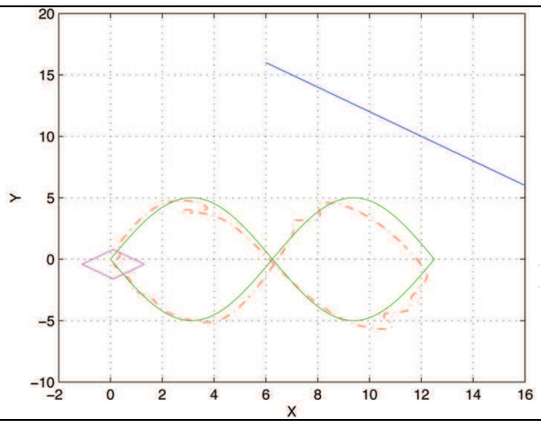

Fig. 5. (a) The trajectory of the mobile robot (dashed line) tracks the reference eight-shaped path (continuous line) when the robot's state vector is estimated with the use of Extended Kalman Filtering.

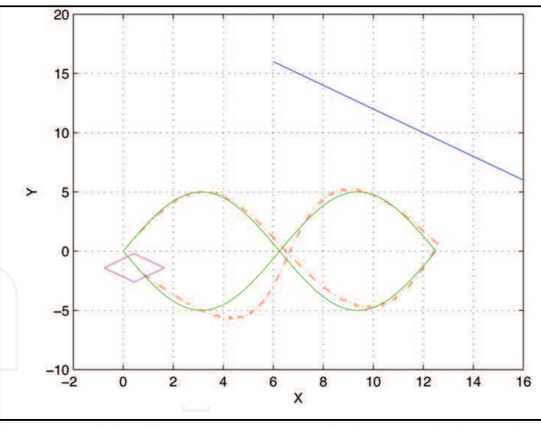

Fig. 5. (b) The trajectory of the mobile robot (dashed line) tracks the reference eight-shaped path (continuous line) when the robot's state vector is estimated with the use of Particle Filtering.

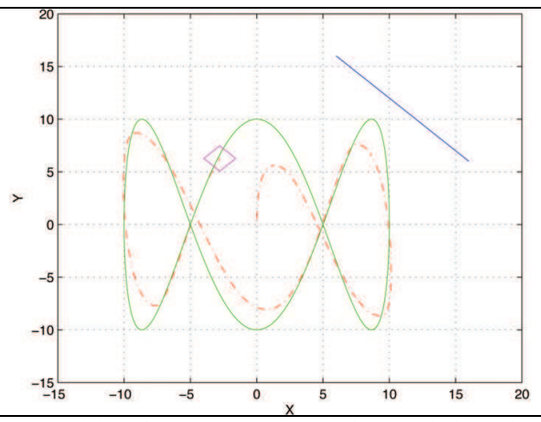

Fig. 6. (a) The trajectory of the mobile robot (dashed line) tracks the reference curve-shaped path (continuous line) when the robot's state vector is estimated with the use of Extended Kalman Filtering.

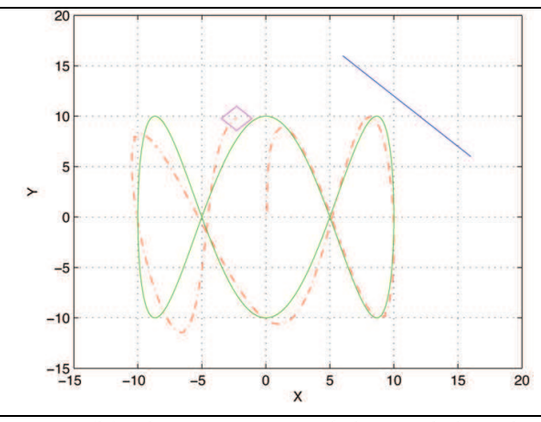

Fig. 6. (b) The trajectory of the mobile robot (dashed line) tracks the reference curve-shaped path (continuous line) when the robot's state vector is estimated with the use of Particle Filtering.

6. Conclusions

The paper has studied flatness-based control and sensor fusion for motion control of autonomous mobile robots. Flatness-based control stems from the concept of differential flatness, i.e. of the ability to express the system parameters (such as the elements of the state vector) and the control input as relations of a function y called *flat output* and of its higher order derivatives. Flatness-based control affects the dynamics of the system in a way similar to control through feedback-linearization. This means that writing the system variables and the control input as functions of the flat output enables transformation of the system dynamics into a linear ODE and subsequently permits trajectory tracking using linear control methods. For linear systems differential-flatness coincides with the property of controllability. Flatness-

based control is applicable to finite dimensional systems (linear or nonlinear) as well as to infinite dimensional systems, such as the ones usually described by PDE.
The problem of motion control of the mobile robots becomes more complicated when the robot's state vector is not directly measurable but has to be reconstructed with the use of measurements coming from distributed sensors. Consequently, the control input generated by the flatness-based control algorithm has to use the estimated state vector of the robotic vehicle instead of the real one. Extended Kalman and Particle filtering have been tested in the problem of estimation of the state vector of a mobile robot through the fusion of position measurements coming from odometric and sonar sensors. The paper has summarized the basics of the Extended Kalman Filter, which is the most popular approach to implement sensor fusion in nonlinear systems. The EKF is a linearization technique, based of a first-order Taylor expansion of the nonlinear state functions and the nonlinear measurement functions of the state model. In the EKF, the state distribution is approximated by a Gaussian random variable. Although the EKF is a fast algorithm, the underlying series approximations can lead to poor representations of the nonlinear functions and the associated probability distributions. As a result, the EKF can sometimes be divergent.
To overcome these weekness of the EKF as well as the constraint of the Gaussian state distribution, particle filtering has been introduced. Whereas the EKF makes a Gaussian assumption to simplify the optimal recursive state estimation, the particle filter makes no assumptions on the forms of the state vector and measurement probability densities. In the particle filter, a set of weighted particles (state vector estimates evolving in parallel) is used to approximate the posterior distribution of the state vector. An iteration of the particle filter includes particle update and weights update. To succeed the convergence of the algorithm at each iteration resampling takes place through which particles with low weights are substituted by particles of high weights.
Simulation tests have been carried out to evaluate the performance of flatness-based control for the autonomous mobile robot, when using the EKF and the particle filter for the localization of the robotic vehicle (through the fusion of measurements coming from distributed sensors). It has been shown, that comparing to EKF, the PF algorithm results in better estimates of the mobile robot's state vector as the number of particles increases, but on the expense of higher computational effort. Consequently, the flatness-based controller which used the robot's state vector coming from the particle filter, had better tracking performance than the flatness-based controller which used the robot's state vector that was estimated by the EKF. It has been also observed that the accuracy in the localization of the mobile robot, succeeded by the particle filter algorithm depends on the number of particles and their initialization.

7. References

Arulampalam, S.; Maskell, S.R., Gordon, N.J. & Clapp, T. (2002). A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, Vol. 50, pp. 174-188.

Campillo, F. (2006). *Particulaire & Modèles de Markov Cachés, Master Course Notes Filtrage et traitment des donées*, Université de Sud-Toulon Var, France.

Caron, F.; Davy,M., Duflos, E. & Vanheeghe, P. (2007). Particle Filtering for Multi-Sensor Data Fusion with Switching Observation Models: Applications to Land Vehicle Positioning. *IEEE Transactions on Signal Processing*, Vol. 55, No. 6, pp., 2703-2719.

Crisan, D. & Doucet, A. (2002). A Survey of Convergence Results on Particle Filtering Methods for Practitioners. *IEEE Transactions on Signal Processing*, Vol. 50, No. 3, pp.736-746.

Fliess, M. & Mounier, H. (1999). Tracking control and π-freeness of infinite dimensional linear systems, *Dynamical Systems, Control, Coding and Computer Vision*, G. Picci and D.S. Gilliam (Eds.), Vol. 258, pp. 41-68, Birkhaüser.

Jetto, L.; Longhi, S. & Venturini, G. (1999). Development and Experimental Validation of an Adaptive Extended Kalman Filter for the Localization of Mobile Robots. *IEEE Transactions on Robotics and Automation*, Vol. 15, No.2.

Kitagawa, G. (1996). Monte-Carlo filter and smoother for non-Gaussian non-linear state-space models. *J. Computat. Graph. Statist*., Vol.5, No.1, pp. 1-25.

Laroche,B. ; Martin, P. & Petit,N. (2007). *Commande par platitude: Equations différentielles ordinaires et aux derivées partielles*, Ecole Nationale Supérieure des Techniques Avancées, Paris.

Lévine, J.; Nguyen, D.V. (2003). Flat output characterization for linear systems using polynomial matrices. *Systems & Control Letters*, Elsevier, Vol. 48, pp. 69-75.

Martin, P. & Rouchon,P. (1999). Systèmes plats: planification et suivi des trajectoires, *Journées X-UPS,Ecole des Mines de Paris*, Centre Automatique et Systèmes, Mai 1999.

Meurer T.; & Zeitz, M. (2004). A modal approach to flatness-based control of flexible structures. *PAMM Proceedings on Applied Mathematics and Mechanics*, Vol. 4, pp. 133-134.

Mounier, H. & Rudolph, J. (2001). Trajectory tracking for π-flat nonlinear delay systems with a motor example, *Nonlinear control in the year 2000* (A. Isidori, F. Lamnabhi-Lagarrigue and W. Respondek, editors), Vol.1, Lecture Notes in Control and Inform. Sci.,vol. 258, pp. 339-352, Springer.

Oriolo,G.; De Luca, A. & Vendittelli, M. (2002). WMR Control Via Dynamic Feedback Linearization: Design, Implementation and Experimental Validation. *IEEE Transactions on Control Systems Technology*, Vol. 10, No.6 , pp. 835-852.

Rigatos, G.G. (2003). Fuzzy Stochastic Automata for Intelligent Vehicle Control. *IEEE Transactions on Industrial Electronics*, Vol. 50, No. 1, pp. 76-79.

Rigatos,G.G.; Tzafestas, S.G. & Evangelidis, G.A. (2001). Reactive parking control of a nonholonomic vehicle via a Fuzzy Learning Automaton. *IEE Proceedings : Control Theory and Applications*, Vol. 148, pp. 169-179.

Rigatos,G.G. (2008). Coordinated motion of autonomous vehicles with the use of a distributed gradient algorithm, *Journal of Applied Mathematics and Computation*, *Elsevier*, Vol 199, No. 2, pp 494-503.

Rigatos, G.G. (2007a). Particle Filtering for state estimation in nonlinear industrial systems, *2nd IC-EpsMsO 2007*, *2nd International Conference on Experiments / Process / System Modelling / Simulation & Optimization,* Athens, Greece, July 2007 .

Rigatos, G.G. & Tzafestas,S.G. (2007). Extended Kalman Filtering for Fuzzy Modeling and Mutli-Sensor Fusion. *Mathematical and Computer Modeling of Dynamical Systems*, Vol. 13, No. 3, Taylor and Francis.

Rigatos,G.G. (2007b). Extended Kalman and particle filtering for sensor fusion in mobile robot localization, *PhysCon 2007, International Conference on Physics and Control,* Potsdam, Germany, Sep. 2007.

Rouchon, P. (2005). Flatness-based control of oscillators. *ZAMM Zeitschrift fur Angewandte Mathematik und Mechanik*, Vol. 85, No.6, pp. 411-421.

Rudolph,J. (2003). *Flatness Based Control of Distributed Parameter Systems*, Steuerungs und Regelungstechnik, Shaker Verlag, Aachen.

Yang, N.; Tian, W.F., Jin, Z.H. & Zhang, C.B. (2005). Particle Filter for sensor fusion in a land vehicle navigation system. *Measurement Science and Technology*, Institute of Physics Publishing, Vol. 16, pp. 677-681.

Thrun, S.; Burgard, W. & Fox, D. (2005). *Probabilistic Robotics*, MIT Press.

Robotics Automation and Control

Edited by Pavla Pecherkova, Miroslav Flidr and Jindrich Dunik

ISBN 978-953-7619-18-3
Hard cover, 494 pages
Publisher InTech
Published online 01, October, 2008
Published in print edition October, 2008

This book was conceived as a gathering place of new ideas from academia, industry, research and practice in the fields of robotics, automation and control. The aim of the book was to point out interactions among various fields of interests in spite of diversity and narrow specializations which prevail in the current research. The common denominator of all included chapters appears to be a synergy of various specializations. This synergy yields deeper understanding of the treated problems. Each new approach applied to a particular problem can enrich and inspire improvements of already established approaches to the problem.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Gerasimos G. Rigatos (2008). Autonomous Robot Navigation Using Flatness-based Control and Multi-Sensor Fusion, Robotics Automation and Control, Pavla Pecherkova, Miroslav Flidr and Jindrich Dunik (Ed.), ISBN: 978-953-7619-18-3, InTech, Available from: http://www.intechopen.com/books/robotics_automation_and_control/autonomous_robot_navigation_using_flatness-based_control_and_multi-sensor_fusion

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2008 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen