Tracking Control and Adaptive Local Navigation for Nonholonomic Mobile Robots

Alexander Mojaev Andreas Zell
University of Tuebingen, Computer Science Dept.,
Computer Architecture,
Sand 1, D - 72076 Tuebingen, Germany
{mojaev,zell}@informatik.uni-tuebingen.de

Abstract. A reactive tracking control law and a flexible local navigation method for non-holonomic mobile vehicles are presented. The tracking controller enables noise tolerant trajectory following and collision avoidance with a feedback control loop through raw sensor data (laser scanner, sonars, IR). The controller needs no knowledge about absolute robot coordinates. Based on the tracking control a local navigation method is described, which allows collision avoidance and flexible path search in a changing environment.

1 Introduction

A reliable motion control is indispensable for developing an autonomous mobile system. When looking at biological systems one can find out that they can move quickly and without risk of collision by perception of relative distance data. This property as well as the fact that special sensors or time-consuming algorithms for self-localisation are needed in order to determine absolute coordinates and the heading angle of a mobile system make it necessary to develop a quick and reactive motion control exclusively by up-to-date sensor information.

In order to solve this problem a reactive and noise tolerant motion controller was designed which makes it possible to cope with dynamic obstacles, and which is robust to noisy sensor data at the same time. The principle of the designed law is to control motions along a given trajectory. One of the advantages of the control is the minimum of input information: only the current, relative deviation from the trajectory (lateral distance) is needed as input value. For that reason it is not necessary to plan the whole motion sequence in advance. The path is able to adapt itself reactively to the dynamic environment.

In order to make it possible to apply noisy sensor data a filtering of the input data was implemented which reduces the influence of the disturbances on the control. Laser scanners, ultrasonic sensors and infrared sensors have turned out to be successful in experiments on a reactive motion control. Even in case of very noisy sensor data such as ultrasonic or infrared sensor data a collision-free trajectory is achieved.

2 Previous Work

A relative tracking along a given trajectory using sliding-mode techniques is described in Balluchi et al [1]. The variable-structure control law for vehicle orientation stabilizes a vehicle moving with approximately constant velocity on the given path. The controller needs

a lateral distance from the path, heading angle error and sign of the curvature of the reference path. However, the last two input values are difficult to extract from sensor data when a tracking trajectory is unknown. Fox, Burgard and Thrun presented a ”dynamic window approach” in [2]. A space of translational and rotational velocities for a robot equipped with a synchro-drive is built depending on the environment. A collision free path is generated by searching for admissible velocities (dynamic window). A time varying control law for solving the tracking problem is described in ([5], Tarín et al). The position control method allows to drive the mobile system from any initial to any final state configuration while minimizing the input energy. However, the control needs global robot coordinates and thus a time consuming localization. Furthermore, no dynamic collision avoidance is implemented. Tsoularis and Kambhampati presented a geometric solution for obstacle avoidance [6]. In consideration of bounds of robot parameters a collision-free path is generated using admissible acceleration and deceleration. Behavior-based navigation using hierarchical neural networks is described in [4] (Pauly). A complex robot task is divided into several behaviors (collision avoidance, wall following, search for free space) that are realized by different neural networks.

3 Synchronous drive kinematics

Our mobile robots of type RWI B-21, named ”Robin” and ”Colin”, are equipped with a four wheel synchronous drive. These robots are controlled by means of an input of translational and rotational velocities v, ω and the corresponding accelerations a, α. The non-holonomic kinematics of such a system can be described by the differential equations (1).

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

We define the complete state vector as $\vec{S} = (\theta, x, y)^T$. The initial state $\vec{S}_0$ must be known in order to calculate the dynamic behaviour of the system.

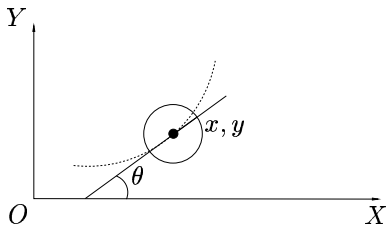

Figure 1: The robot in the absolute Cartesian coordinate system.

4 Differential drive kinematics

Our small robots of type Active Media Pioneer 2 DX are equipped with a differential drive kinematics where the wheels are driven by different motors.

$$\begin{cases} \dot{\theta} = \dfrac{R}{D}(\omega_r - \omega_l) \\ \dot{x} = \dfrac{R(\omega_r + \omega_l)}{2}\,\cos\theta \\ \dot{y} = \dfrac{R(\omega_r + \omega_l)}{2}\,\sin\theta \end{cases} \tag{2}$$

where R is the radius of a wheel and D is the distance between the wheels.

By means of the following equations the wheel velocities ω_l and ω_r can be converted into the robot-centered translational and rotational velocities (3) and vice versa (4): The further

$$\begin{cases} \omega = \dfrac{R}{D}(\omega_r - \omega_l) \\ v = \dfrac{R(\omega_r + \omega_l)}{2} \end{cases} \quad (3) \qquad \begin{cases} \omega_r = \dfrac{1}{R}(v + \dfrac{D\,w}{2}) \\ \omega_l = \dfrac{1}{R}(v - \dfrac{D\,w}{2}) \end{cases} \quad (4)$$

calculations are developed in general form for the synchronous drive kinematics (1). For the differential drive the results can be transformed by (4).

5 Control of the robot along a given trajectory

A non-holonomous vehicle cannot be controlled by a direct input of goal coordinates. For that reason various control methods are used in order to make it possible for the robot to reach its target. Many position control methods solve this problem by using the odometry information. Due to incorrect odometry data these processes can only be applied to a limited extent in the real environment. Therefore a position control law was developed which needs only the relative deviation from a given trajectory as a control input. The cycle is closed by the loop "actuators-environment-sensors" which makes reactive control possible [3].

5.1 Tracking with relative coordinates

A continuous asymptotic convergence to the trajectory can be achieved by the following exponential tracking function

$$y' = y'_0 \exp\left(-K_{\text{trk}}\, x'\right) \quad (5)$$

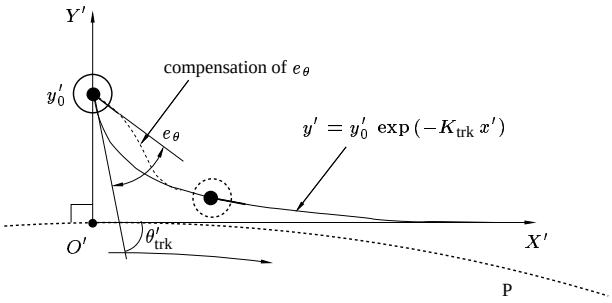

Figure 2: The tracking function and convergence of the robot to the trajectory.

so that the curvature and thus the convergence speed can be modified with the coefficient K_{trk}. A coordinate system $(X'O'Y')$ aligned with the given trajecory P (Fig. 2) is defined. The system moves along the trajectory so that the abscissa $O'X'$ always results in a tangent. In this coordinate system a tracking trajectory $y' = f_{\text{trk}}(x')$ is defined which makes a convergence with the given trajectory P possible.

By differentiation of (5) the derivation

$$\frac{dy'}{dx'} = -K_{\text{trk}}\, y' \quad (6)$$

and the tangent inclination θ'_{trk} are calculated:

$$\theta'_{\text{trk}} = \arctan\left(\frac{dy'}{dx'}\right) = \arctan\left(-K_{\text{trk}}\, y'\right) \quad (7)$$

Note that the derivation (6) and the tangent inclination (7) do not directly depend on the time but only on the lateral distance of the tracking path and thus the tracking curve is independent on the translational velocity. The first relative control equation now results from (7):

$$\omega = \frac{d}{dt}\big(\arctan\left(-K_{\text{trk}}\, r\right)\big) \tag{8}$$

Here r denotes the input parameter of the control. It is the deviation of the robot from the trajectory $r = y'$ and is determined by sensors in real time. One condition of the control law (8) is that the orientation of the robot (heading angle θ') must be identical with the tangent inclination θ'_{trk}. Because this condition can not be satisfied, we introduce a compensation of the robot heading angle error (see Fig. 2):

$$e_\theta = \theta'_{\text{trk}} - \theta' \tag{9}$$

This error can be eliminated by the additional rotation with the angular velocity:

$$\omega_{\text{comp}} = K_{\text{comp}}\, e_\theta \tag{10}$$

where $0 < K_{\text{comp}} < 1$ is the compensation parameter. The lower K_{comp} is, the longer the angle compensation takes. The whole control equation arises from (8) in consideration of (10) as follows:

$$\omega = \frac{d}{dt}\big(\arctan\left(-K_{\text{trk}}\, r\right)\big) + K_{\text{comp}}\,\left(\arctan\left(-K_{\text{trk}}\, r\right) - \hat{\theta}'\right) \tag{11}$$

An estimation of robot's relative orientation $\hat{\theta}'$ with reference to the moving coordinate system $X'O'Y'$ can be obtained either indirectly through

$$\hat{\theta}' = \arcsin\left(\frac{\dot{r}}{v}\right) \tag{12}$$

or directly from sensor data. In Fig. 3 the tracking trajectories and errors with various initial orientation angles are shown. The described tracking (11) works even without the estimation of θ' (Fig. 3, bottom). In case of $\theta' = 0$ the equation (11) corresponds to a PI control in which the estimation of the orientation error is realized by the integration of θ'_{trk}.

The developed control possesses the following advantages:

- no knowledge about absolute robot coordinates is needed. The input of the control is the actual lateral distance from the trajectory.
- reactive tracking. We can continuosly adjust the trajectory to the environment.
- smooth exponential convergence to the trajectory.

5.2 Corridor following and collision avoidance

We applied this tracking technique to a corridor following problem. Solving this task by our reactive motion control we can automatically avoid all obstructions if we define the given trajectory as the middle path. The input of the control in the corridor following task is

$$r = \frac{R_{\text{right}} - R_{\text{left}}}{2} \tag{13}$$

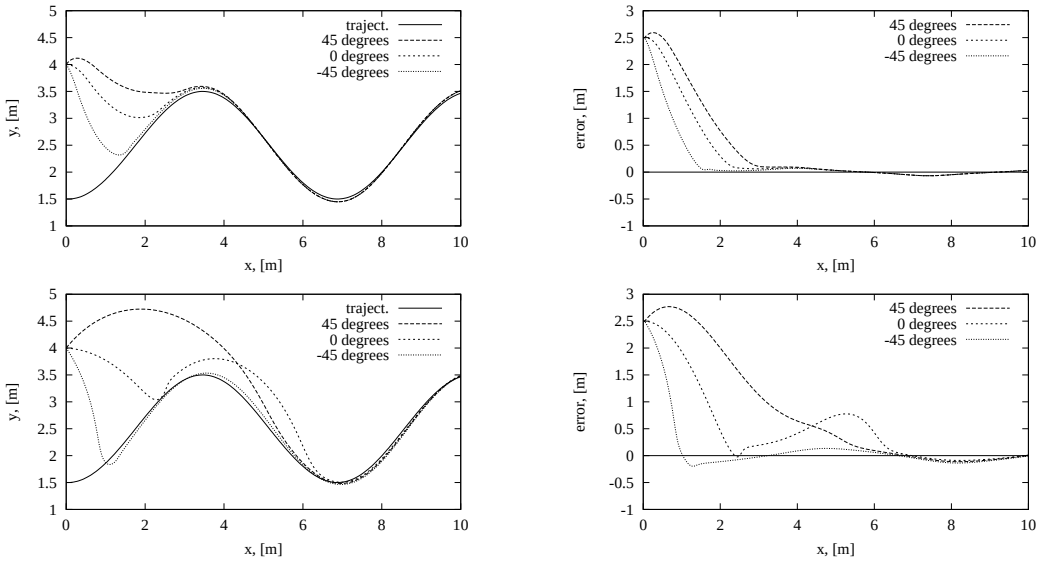

Figure 3: The tracking along a sinusoidal function for three different initial robot orientations (-45, 0 and 45 deg.) Top: with θ' estimation by (12) ($K_{\text{comp}} = 0.7$), bottom: without θ' estimation ($\theta' = 0, K_{\text{comp}} = 0.12$). Left are shown the tracking trajectories, right the corresponding control errors. Other parameters: K_{trk}=7, v=0.7 m/s; limits of the maximal rotation acceleration ($\alpha_{\max}$ = 5 rad/s^2) and velocity ($\omega_{\max}$ = 2.9 rad/s) were also taken into account.

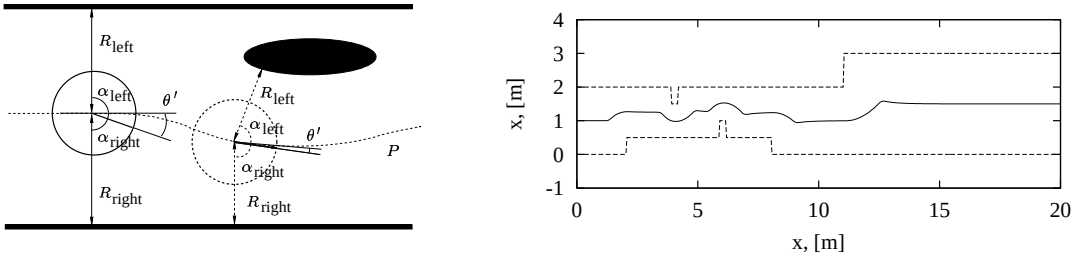

Figure 4: Corridor following: trajectory definition (left) and simulated tracking (right) v=0.5 m/s.

where R_{right}, R_{left} the left and right distance to the obstacles regarding the robot orientation (Fig. 4, left). Motion control is realized by (11). The heading angle error can either be neglected (if no angle measurements exist) or calculated from sensor data:

$$\hat{\theta}' = \frac{\alpha_{\text{right}} - \alpha_{\text{left}}}{2} \tag{14}$$

The simulation results with no knowledge about heading angle ($\hat{\theta}' = 0, \theta_0 = 0$) are shown in Fig. 4, right.

5.3 Data filtering

The control input r is determined from sensor data and contains noise. This noise component is intensified by the differential equation part (11) (because its spectrum contains mostly high frequencies) and the mobile system can possibly lose control. We use two filtering methods: a simple non-linear filtering based on the observation that the distances derivatives $\dot{R}_{\text{right}}$ and $\dot{R}_{\text{left}}$ are very rarely greater than the robot's translational velocity $\dot{R}_{\text{right}}, \dot{R}_{\text{left}} \leq v$. Larger values

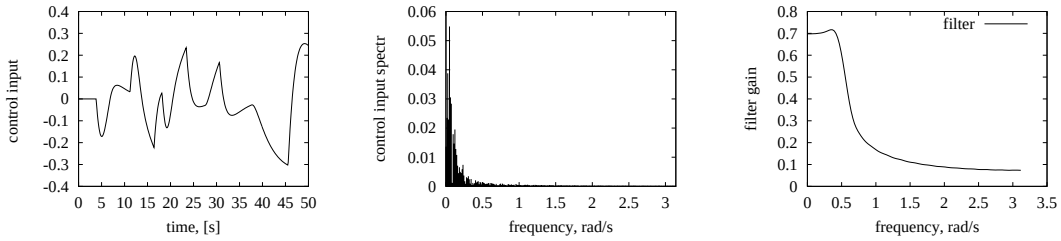

Figure 5: The control input r (example in Fig. 4), its Fourier spectrum and the frequency characteristics of the filter $h(n)$ with $N = 32,\ \omega_c = 0.5$.

are interpreted as erroneus and ignored:

$$\begin{aligned} R_{\text{right}}(t) &= R_{\text{right}}(t - \Delta t) + \min\big(\Delta R_{\text{right}}(t),\ v\,\Delta t\big) \\ R_{\text{left}}(t) &= R_{\text{left}}(t - \Delta t) + \min\big(\Delta R_{\text{left}}(t),\ v\,\Delta t\big) \end{aligned} \tag{15}$$

Additionally we use a linear filtering: if we analyse the Fourier spectra of the control input, we can observe that the signal has mostly low frequency components in contrast to noise (Fig. 5). So we process the input control by a low pass filter (FIR) with minimal time delay to minimize sensor noise. The impulse characteristics is

$$h(n) = h_0(n)\, w_{\text{ham}}, \quad n \in [0, N]. \tag{16}$$

Here

$$h_0(n) = \frac{\sin(\omega_c(n-1))}{\pi(n-1)}, \qquad w_{\text{ham}}(n) = \beta + (1-\beta)\cos\left(\frac{\pi n}{N}\right) \tag{17}$$

are the impulse characteristics of an ideal low pass filter with N coefficients and the hamming window with a parameter $\beta = 0.54$, ω_c is the filter cut frequency. We do not use the coefficients with $n < 0$ to avoid a signal delay. Although it implicates a degradation in filter efficiency, the noise damping in the range $\omega > \omega_c$ is sufficient to achieve a relevant improvement of the control with very noisy sensor data.

The filtering is realized as a convolution of the control input with impulse characteristics:

$$\hat{r}_i = \sum_{n=0}^{N} h(n)\, r(i-n) \tag{18}$$

Fig. 6 demonstrates the control improvement in a real environment using laser scanner data: unnecessary rotation fluctuations are eliminated.

5.4 Control of translational velocity

The robot's translational velocity is determined from the robot's free front D and side $R = \min(R_{\text{right}}, R_{\text{left}})$ distances:

$$v = v_{\min} + (v_{\max} - v_{\min})\frac{R - R_{\min}}{R}\,\frac{D - D_{\min}}{D} \tag{19}$$

where $R_{\min}$ and $D_{\min}$ are the safety radius and the frontal safety distance.

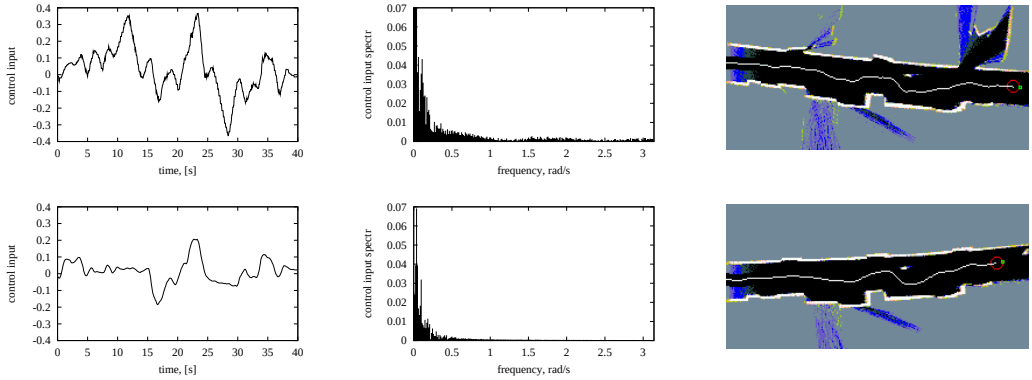

Figure 6: Experiments in a real indoor environment: control quality improvement dealing with noisy sensor data by filtering, from left to right: the control input r, its Fourier spectrum S_r, and robot path without filtering. Right: the control input processed with the filters described.

5.5 The virtual corridor

The "virtual corridor" is a set of points building a closed space within which the robot can move freely. This corridor connects the start position of the robot $(x_\mathrm{s}, y_\mathrm{s})^T$ with the local navigation $(x_\mathrm{t}, y_\mathrm{t})^T$ target (Fig. 7). The virtual corridor is modified during the robot motion: if the target distance decreases, the rear line of the virtual corridor moves forwards, leaving sufficient space for possible alternative target searching, however. The wider the corridor is, the more space the robot has for collision avoidance, although the navigation can take longer. The width D and length (varying R_b and (R_f) of the virtual corridor are chosen depending on the environment properties. The nodes' coordinates are calculated as follows:

$$\begin{aligned}
\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} &= \begin{pmatrix} x_\mathrm{s} \\ y_\mathrm{s} \end{pmatrix} + R_\varphi \begin{pmatrix} -0.5D \\ -R_\mathrm{b} \end{pmatrix} \\
\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} &= \begin{pmatrix} x_\mathrm{s} \\ y_\mathrm{s} \end{pmatrix} + R_\varphi \begin{pmatrix} -0.5D \\ d_\mathrm{t} + R_\mathrm{f} \end{pmatrix} \\
\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} &= \begin{pmatrix} x_\mathrm{s} \\ y_\mathrm{s} \end{pmatrix} + R_\varphi \begin{pmatrix} 0.5D \\ d_\mathrm{t} + R_\mathrm{f} \end{pmatrix} \\
\begin{pmatrix} x_3 \\ y_3 \end{pmatrix} &= \begin{pmatrix} x_\mathrm{s} \\ y_\mathrm{s} \end{pmatrix} + R_\varphi \begin{pmatrix} 0.5D \\ -R_\mathrm{b} \end{pmatrix},
\end{aligned} \tag{20}$$

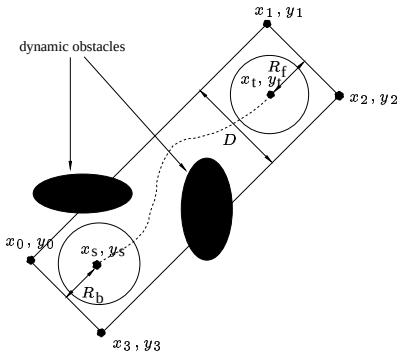

Figure 7: Points of the virtual corridor.

where

$$R_\varphi = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix}, \varphi = \arctan\left(\frac{y_\mathrm{t} - y_\mathrm{s}}{x_\mathrm{t} - x_\mathrm{s}}\right) \quad \text{and} \quad d_\mathrm{t} = \sqrt{(x_\mathrm{t} - x_\mathrm{s})^2 + (y_\mathrm{t} - y_\mathrm{s})^2}.$$

The corridor is first built in the robot centered coordinate system and then rotated in the target direction. The virtual distances $R_{\text{vc right}}$ and $R_{\text{vc left}}$ which are needed for motion control, are calculated by well known trigonometry rules.

5.6 Dead end coverage

Using a radial sensor data we can determine maximal collision-free distances for each direction. If these distances in the ± 90 deg. sector are lower than the frontal safety distance, the

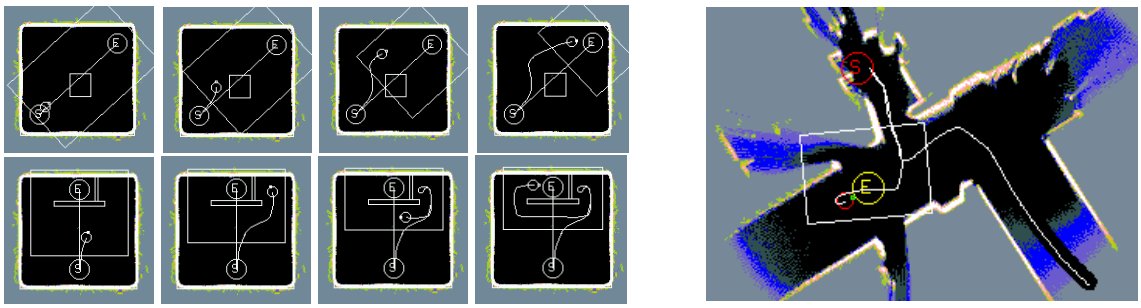

Figure 8: Adaptive local navigation by the ”virtual corridor” technique, left: some simulation examples, right: experimental navigation in a real world environment. The robot must reach the target points “S” and “E”.

robot interprets the situation as dead end and rotates 180 degrees.

5.7 Fusion of virtual and real environment

A virtual corridor is always built in relation to a real environment, which is represented by a grid map model. This environment model and the robot coordinates are continuosly updated by a global localization algorithm. We integrate the real obstacles into our virtual corridor, modifying left and right robot side distances used for tracking control (22). The examples of flexible local navigation in a changing environment are shown in Fig. 8.

$$R_{\text{right}} = \min_{i=0}^{N_{\text{sensors}}} R_{\text{right}_i}, \quad R_{\text{left}} = \min_{i=0}^{N_{\text{sensors}}} R_{\text{left}_i}$$

6 Experimental results

The described approaches to tracking control and local navigation with sonars and a laser-scanner in indoor environments have been implemented on two mobile robots ”Robin” and ”Colin” of type RWI-B21. At an exploration velocity up to the maximal robot velocity (1 m/s using laser scanner, 0.8 m/s using only sonar data) the collision avoidance, navigation and mapping are executed in real-time. The experiments show high flexibility of the described navigation and collision avoidance technique (Fig. 8, right) in a complex indoor environment.

References

[1] A. Balluchi, A. Bicchi, A. Balestrino, G. Casalino, Path Tracking Control for Dubin’s Cars, Proc. IEEE Int. Conf. on Robotics and Automation, (1996).

[2] D. Fox, W. Burgard, S. Thrun, The Dynamic Window Approach to Collision Avoidance, IEEE Robotics & Automation Magazine, Vol. 4, Num. 1, 1–100, (1997).

[3] A. Mojaev, Umgebungswahrnehmung, Selbstlokalisation und Navigation mit einem mobilen Roboter, Ph.D. thesis, University of Tbingen, WSI, Shaker Verlag ISBN 3-8265-7865-1, (2001).

[4] M. Pauly, Exploration und Navigation mit hierarchischen Neuronalen Netzen, Autonome Mobile Systeme, ’96, G. Schmidt, U. Hanebeck and F. Freyberger (eds.), 180–189, (1996).

[5] C. Tarín, H. Brugger, B. Tibken, E.P. Hofer, Optimal Control for a Synchronous Driven Unicycle-Like Autonomous mobile Robot, Autonome Mobile Systeme, ’99, G. Schmidt, U. Hanebeck and F. Freyberger (eds.), 313–322, (1999).

[6] A. Tsoularis, C. Kambhampati, Avoiding Moving Obstacles by Deviation from a Mobile Robot’s Nominal Path, The International Journal of Robotics Research, Num. 5, Vol. 18, 454–465, May, (1999).