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

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

154
Countries delivered to

Our authors are among the

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

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

LQR Control Methods for Trajectory Execution in Omnidirectional Mobile Robots

Luis F. Lupián and Josué R. Rabadán-Martin
Mobile Robotics & Automated Systems Lab, Universidad La Salle
Mexico

1. Introduction

Omnidirectional mobile robots present the advantage of being able to move in any direction without having to rotate around the vertical axis first. While simple straight-line paths are relatively easy to achieve on this kind of robots, in many highly dynamic applications straight-line paths might just not be a feasible solution. This may be the case because of two main reasons: (1) there may be static and moving obstacles between the initial and desired final position of the robot, and (2) the dynamic effects of the inertia of the robot may force it to execute a curved path. This chapter will address these two situations and present a segment-wise optimal solution for the path execution problem which is based on a Linear-Quadratic Regulator.

It must be emphasized that, rather than attempting to perform an exact path tracking, the approach presented here deals with the problem of visiting a sequence of target circular regions without specifying the path that will connect them. The freedom given to the connecting paths brings the opportunity for optimization. In fact, the path that the robot will take from one circular region to the next will emerge as the solution of an optimal control problem, hence the term *segment-wise optimal* solution.

Omnidirectional wheels have the property of sliding laterally with almost zero force while providing full traction in the rolling direction. This effect is achieved by adding a set of smaller wheels around the periphery of the main wheel, as depicted in Fig. 1.

Fig. 1. Omnidirectional wheel

By using several omnidirectional wheels distributed around the periphery of a cylindrical robot one can achieve the effect of driving the robot in a direction dependent on a vector sum of forces. This idea is illustrated in Fig. 2. The forces applied at the wheels by the motors give translational and rotational movement to the mobile robot.
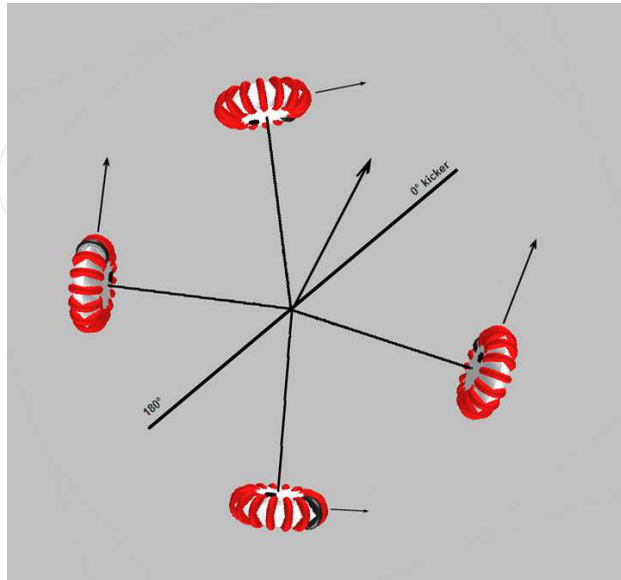

Fig. 2. Omnidirectional drive

Omnidirectional movement on a mobile robot has many applications, and thus has received attention from the scientific community over several years. Borenstein and Evans developed a control strategy that allows trajectory control on a mobile robot that uses conventional non-omnidirectional wheels and thus requires the drive wheels to rotate with respect to the vertical axis (Borenstein & Evans, 1997). With regard to omnidirectional mobile robots based on omnidirectional wheels there are several works that deserve attention. Ashmore and Barnes presented a detailed analysis of the kinematics of this kind of mobile robots and show that under certain circumstances a curved path may be faster than the straight line between two points (Ashmore & Barnes, 2002). Balkcom et al. characterized time-optimal trajectories for omnidirectional mobile robots (Balkcom et al., 2006a;b). Kalmár-Nagy et al. developed a control method to generate minimum-time trajectories for omnidirectional mobile robots (Kalmár-Nagy et al., 2004; 2002), then Purwin and D'Andrea presented the results of applying this method to a RoboCup F180 omnidirectional mobile robot (Purwin & D'Andrea, 2005).
Section 2 deals with the first necessary step towards applying LQR control for trajectory execution, which is formulating a state-space model for the dynamics of the omnidirectional mobile robot (Lupián & Rabadán-Martin, 2009).
This state-space model is non linear with respect to the control due to the fact that the robot rotates around the vertical axis and the pan angle is one of the state variables. In Sec. 3 of this chapter we show how to overcome this problem to successfully apply a Linear Quadratic Regulator for the case of three, four and five-wheeled omnidirectional mobile robots. In Sec. 4 we present a method to generate a segment-wise optimal path by solving an LQR control problem for each segment between the initial state through a sequence of target regions that ends at the desired final state of the omnidirectional mobile robot. Finally, Sec. 5 presents the results of several simulation experiments that apply the methods described in this chapter.

2. State-space dynamic model

The analysis presented pertains to a specific class of omnidirectional mobile robots that are of cylindrical shape with n omnidirectional wheels distributed around the periphery of the body of the robot, with the axes of the wheels intersecting at the geometrical vertical axis of the robot. Figure 3 shows an instance of this class of omnidirectional mobile robots for the case of $n = 5$ omnidirectional wheels. It is not necessary to have a uniform distribution of the wheels around the periphery of the robot. That is, the angle that separates the axis of one wheel to the next does not need to be the same. However, there is an obvious restriction that must be met in order to maintain stability, which is that the projection of the center of mass of the robot onto the ground must be contained within the convex hull of the set of contact points of the n wheels with the ground. For simplicity of the analysis, the mass of the robot is assumed to be distributed uniformly, so the center of mass is contained within the geometrical vertical axis of the robot.
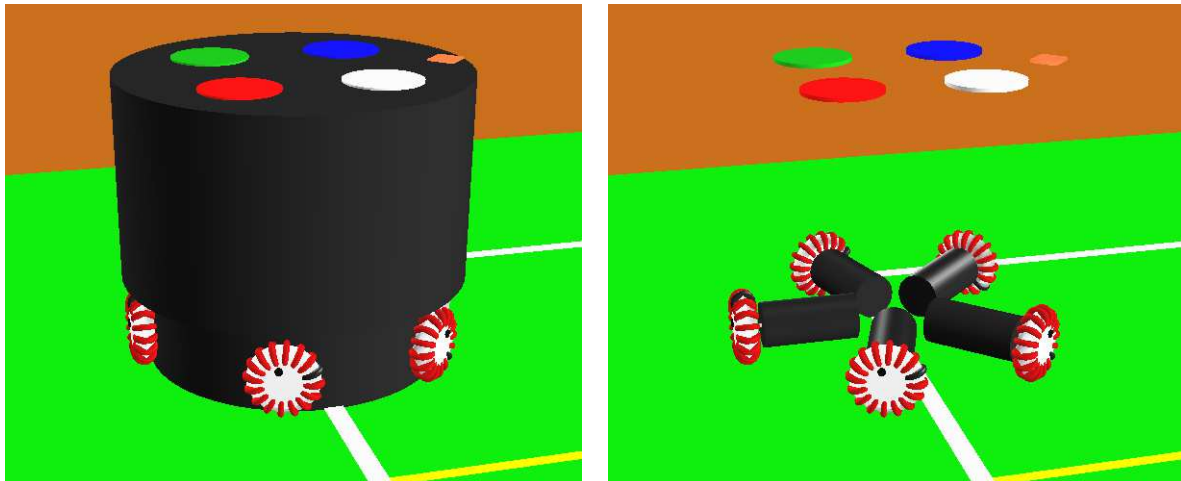

Fig. 3. Omnidirectional robot

In the literature, there are several proposals for the dynamic model of an omnidirectional mobile robot. One of the main problems with these models is that they do not provide a complete state-space representation, so it is not possible to perform state-space control by using one of them. Most of these proposals are based on the force coupling matrix (Gloye & Rojas, 2006), which provides a direct relationship between the torques applied by the driving motors and the accelerations in the x, y and angular directions.
In compact form, these equations may be written as follows

$$
\begin{aligned}
a &= \frac{1}{M}(F_1 + F_2 + \cdots + F_n) \\
\dot{\omega} &= \frac{R}{I}(f_1 + f_2 + \cdots + f_n)
\end{aligned}
\tag{1}
$$

where a is the acceleration of the robot with respect to the inertial reference, $\dot{\omega}$ is the angular acceleration, n is the number of omnidirectional wheels around the periphery of the robot, F_i is the vector force applied by motor i, f_i is the signed scalar value of F_i (positive for counter clock-wise rotation), M is the mass of the robot, R is its radius and I is its moment of inertia.

Taking into account the driving motor distribution shown in Fig. 4, Eq. 1 may be re-written as

$$
\begin{aligned}
Ma_x &= -f_1 \sin\theta_1 - f_2 \sin\theta_2 - \cdots - f_n \sin\theta_n \\
Ma_y &= f_1 \cos\theta_1 + f_2 \cos\theta_2 + \cdots + f_n \cos\theta_n \\
MR\dot{\omega} &= \frac{1}{\alpha}(f_1 + f_2 + \cdots + f_n)
\end{aligned}
\tag{2}
$$

where x and y are the axes of the inertial frame of reference, a_x is the x component of the acceleration, a_y is the y component of the acceleration, α is such that $I = \alpha MR^2$, and θ_i is the angular position of driving motor i with respect to the robot's referential frame as shown in Fig. 4. Equation 2 is expressed in matrix form as

$$
\begin{bmatrix} a_x \\ a_y \\ R\dot{\omega} \end{bmatrix} = \frac{1}{M}
\begin{bmatrix}
-\sin\theta_1 & -\sin\theta_2 & \cdots & -\sin\theta_n \\
\cos\theta_1 & \cos\theta_2 & \cdots & \cos\theta_n \\
\frac{MR}{I} & \frac{MR}{I} & \cdots & \frac{MR}{I}
\end{bmatrix}
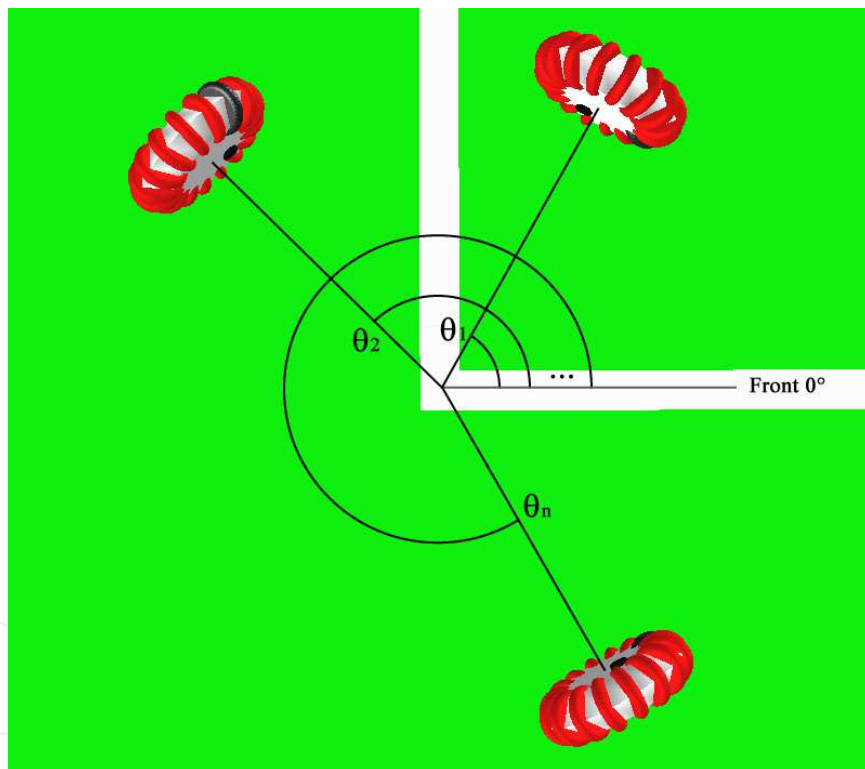\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}
\tag{3}
$$

Fig. 4. Driving motor distribution

Equation 3 gives a general idea about the dynamic behavior of the mobile robot. However, it can not be considered a complete state-space model since it does not deal with the inertial effects of the mass of the robot, so we define a state vector that is complete enough to describe the dynamic behavior of the mobile robot (Lupián & Rabadán-Martin, 2009). We define the state vector z as

$$
z = \begin{bmatrix} x & y & \beta & \dot{x} & \dot{y} & \dot{\beta} & \mu_1 & \cdots & \mu_n \end{bmatrix}^{\mathrm{T}}
\tag{4}
$$

where (x, y) is the position of the robot with respect to the inertial reference, $(\dot{x}, \dot{y})$ is the vector velocity of the robot with respect to the field plane, β is the angular position and $\dot{\beta}$ is

the angular velocity. Those are the six state variables we are interested in controlling, so it would suffice to use them to describe the state of the robot.

However, for simulation purposes we introduce variables μ_i, which correspond to the angular positions of each of the omnidirectional wheels of the robot. The decision to include these state variables was initially motivated by the fact that the dynamic response of the robot would be simulated on a virtual environment developed in OpenGL, and the motion of the wheels would make this simulation more realistic. On the physical robot, however, these state-variables will have a more important role because, unlike the other six state variables, the angular positions of the wheels can be measured directly through motor encoders, so one can implement a state observer to estimate the remaining variables.

In order to express the state equations in a compact and clear form, the state vector will be partitioned as follows:

$$
\begin{aligned}
z_1 &= \begin{bmatrix} x & y & \beta & \dot{x} & \dot{y} & \dot{\beta} \end{bmatrix}^{\mathrm{T}} \\
z_{11} &= \begin{bmatrix} x & y & \beta \end{bmatrix}^{\mathrm{T}} \\
z_{12} &= \begin{bmatrix} \dot{x} & \dot{y} & \dot{\beta} \end{bmatrix}^{\mathrm{T}} \\
z_2 &= \begin{bmatrix} \mu_1 & \dots & \mu_n \end{bmatrix}^{\mathrm{T}}
\end{aligned}
\tag{5}
$$

In terms of z_1 y z_2 the state vector z can be expressed as

$$
z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}
\tag{6}
$$

and the state-space model becomes

$$
\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u
\tag{7}
$$

where u is the control vector composed of the scalar forces f_i divided by the mass of the robot

$$
u = \frac{1}{M} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix},
\tag{8}
$$

and the matrices are defined as follows. Matrix A_{11} simply expresses the relationships among the first six state variables, and is given by

$$
A_{11} = \begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix}
\tag{9}
$$

Matrix B_1 is obtained from Eq. 3 and becomes

$$
B_1 = \begin{bmatrix}
0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 \\
-\sin\theta_1 & -\sin\theta_2 & \cdots & -\sin\theta_n \\
\cos\theta_1 & \cos\theta_2 & \cdots & \cos\theta_n \\
\frac{MR}{I} & \frac{MR}{I} & \cdots & \frac{MR}{I}
\end{bmatrix}
\tag{10}
$$

Matrix B_1 expresses the correct influence of each force over the respective acceleration only for the case in which the angular position β of the robot is zero. However, in order to take into account the fact that the robot will rotate as time goes by this matrix should also depend on β as follows

$$B_1(\beta) = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ -\sin(\theta_1 + \beta) & -\sin(\theta_2 + \beta) & \cdots & -\sin(\theta_n + \beta) \\ \cos(\theta_1 + \beta) & \cos(\theta_2 + \beta) & \cdots & \cos(\theta_n + \beta) \\ \frac{MR}{I} & \frac{MR}{I} & \cdots & \frac{MR}{I} \end{bmatrix}$$

$$B_1(\beta) = \begin{bmatrix} 0_{3 \times n} \\ B_{12}(\beta) \end{bmatrix}$$

(11)

A_{12} and A_{22} are zero matrices of size $6 \times n$ and $n \times n$ respectively. Matrix A_{21} expresses the angular motion of the wheels with respect to the motion of the mobile robot and, like B_1, it is dependent on the angular position β of the robot:

$$A_{21}(\beta) = \frac{1}{r} \begin{bmatrix} 0_{n \times 3} & A_{212}(\beta) \end{bmatrix}$$

$$A_{212}(\beta) = \frac{1}{r} \begin{bmatrix} \sin(\theta_1 + \beta) & -\cos(\theta_1 + \beta) & -R \\ \sin(\theta_2 + \beta) & -\cos(\theta_2 + \beta) & -R \\ \vdots & \vdots & \vdots \\ \sin(\theta_n + \beta) & -\cos(\theta_n + \beta) & -R \end{bmatrix}$$

(12)

where r is the radius of the omnidirectional wheels.

Taking into account that both A_{21} and B_1 depend on β the state model in Eq. 7 may be reformulated as

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & 0_{6 \times n} \\ A_{21}(\beta) & 0_{n \times n} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} B_1(\beta) \\ 0_{n \times n} \end{bmatrix} u$$

(13)

3. Global linearization of the state-space dynamic model

Since the angular position β is one of the state variables this implies that the model in Eq. 13 is non-linear, and that represents an important difficulty for the purpose of controlling the robot. This is why it became necessary to find a way to linearize the model.

The solution to this problem required a shift in perspective in relation to the model. The only reason why the angle β has a non-linear effect on the dynamics of state variables in z_1 is because as the robot rotates also the driving forces, which are the control variables, rotate.

Let $F = F_1 + F_2 + \ldots + F_n$ be the resulting vector force applied by the omnidirectional wheels to the robot when it is at the angular position β and the control vector is u. Let $\check{u}$ be the control vector that will produce the same resulting vector force F when the robot is at the angular position $\beta = 0$. This idea is explained by Fig. 5.

The control vectors u and $\check{u}$ are then related by

$$\begin{bmatrix} \sin(\theta_1) & \cdots & \sin(\theta_n) \end{bmatrix} \check{u} = \begin{bmatrix} \sin(\theta_1 + \beta) & \cdots & \sin(\theta_n + \beta) \end{bmatrix} u$$

$$\begin{bmatrix} \cos(\theta_1) & \cdots & \cos(\theta_n) \end{bmatrix} \check{u} = \begin{bmatrix} \cos(\theta_1 + \beta) & \cdots & \cos(\theta_n + \beta) \end{bmatrix} u$$
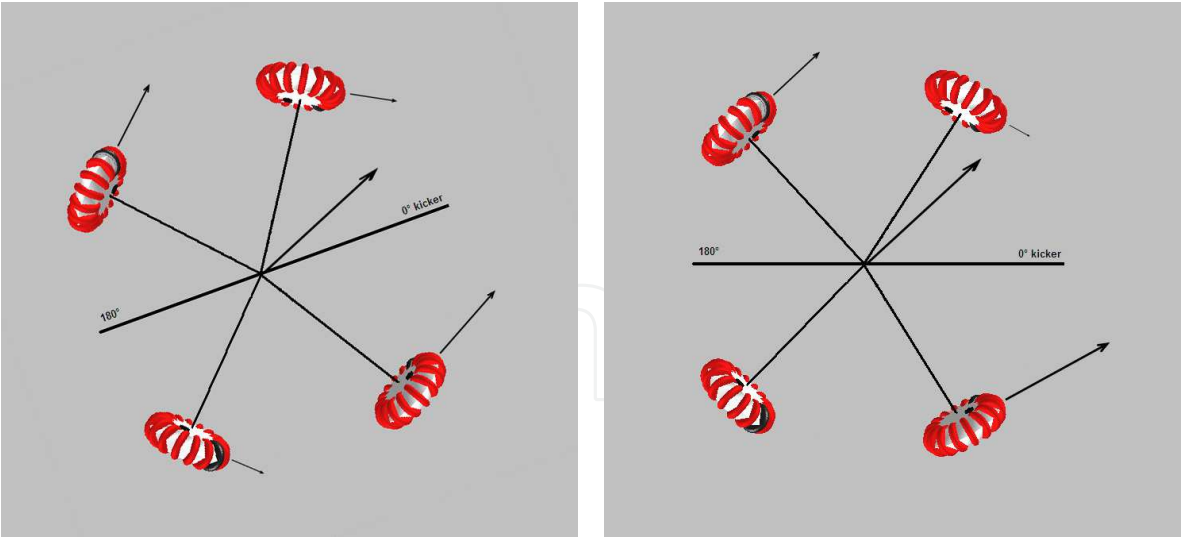
(14)

Fig. 5. Change of variable in control vector

Moreover, in order to ensure that the angular acceleration will remain unchanged it is necessary that the resulting scalar force $f = f_1 + f_2 + \ldots + f_n$ is the same in both cases, which translates to

$$\sum_{i=1}^{n} \breve{u}_i = \sum_{i=1}^{n} u_i \qquad (15)$$

What we need from Eqs. 14 and 15 is a one-to-one mapping that can take us from u to $\breve{u}$ back and forth. Since u is n-dimensional and we have three equations the system is under-determined whenever $n > 3$. Although the pseudo-inverse would provide a linear transformation from one domain to the other, this transformation would be rank-deficient and thus would not be one-to-one. Our solution requires then to add $n - 3$ *complementary equations*. In order to avoid unnecessary numerical problems, these equations should be chosen so that the linear transformation is well-conditioned for any value of the angle β.

3.1 Transformation matrix for $n = 3$

The simplest case comes when the number of omnidirectional wheels of the robot is $n = 3$, see Fig. 6. For this particular case the number of equations provided by Eqs. 14 and 15 is equal to the number of control variables, so there is no need for additional equations to complete the transformation.
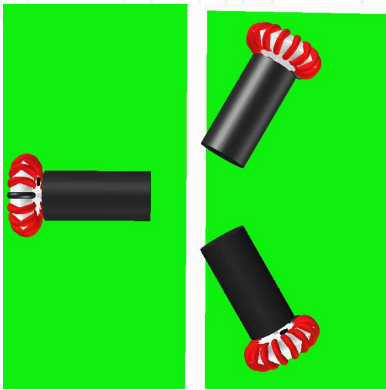

Fig. 6. Omnidirectional drive for $n = 3$

Equations 14 and 15 can be written in matrix form as follows

$$\begin{bmatrix} \sin(\theta_1) & \sin(\theta_2) & \sin(\theta_3) \\ \cos(\theta_1) & \cos(\theta_2) & \cos(\theta_3) \\ 1 & 1 & 1 \end{bmatrix} \breve{u} = \begin{bmatrix} \sin(\theta_1 + \beta) & \sin(\theta_2 + \beta) & \sin(\theta_3 + \beta) \\ \cos(\theta_1 + \beta) & \cos(\theta_2 + \beta) & \cos(\theta_3 + \beta) \\ 1 & 1 & 1 \end{bmatrix} u \quad (16)$$

We can then define the transformation matrix $\Omega_3(\beta)$ according to

$$\Omega_3(\beta) = \begin{bmatrix} \sin(\theta_1) & \sin(\theta_2) & \sin(\theta_3) \\ \cos(\theta_1) & \cos(\theta_2) & \cos(\theta_3) \\ 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} \sin(\theta_1 + \beta) & \sin(\theta_2 + \beta) & \sin(\theta_3 + \beta) \\ \cos(\theta_1 + \beta) & \cos(\theta_2 + \beta) & \cos(\theta_3 + \beta) \\ 1 & 1 & 1 \end{bmatrix} \quad (17)$$

3.2 Transformation matrix for $n = 4$

For the particular case of $n = 4$ (Fig. 7), it is easy to see that Eq. 18 satisfies the requirement of completing a well-conditioned 4×4 transformation since it is orthogonal to two of the other equations and still sufficiently linearly independent from the third one.

$$\begin{bmatrix} 1 & -1 & 1 & -1 \end{bmatrix} \breve{u} = \begin{bmatrix} 1 & -1 & 1 & -1 \end{bmatrix} u \quad (18)$$
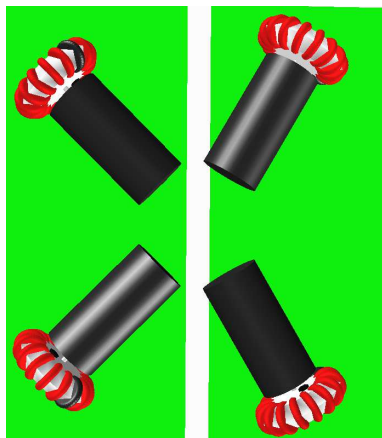

Fig. 7. Omnidirectional drive for $n = 4$

Equations 14, 15 and 18 can be put in matrix form as follows

$$\begin{bmatrix} \sin(\theta_1) & \sin(\theta_2) & \sin(\theta_3) & \sin(\theta_4) \\ \cos(\theta_1) & \cos(\theta_2) & \cos(\theta_3) & \cos(\theta_4) \\ 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \breve{u} =$$

$$\begin{bmatrix} \sin(\theta_1 + \beta) & \sin(\theta_2 + \beta) & \sin(\theta_3 + \beta) & \sin(\theta_4 + \beta) \\ \cos(\theta_1 + \beta) & \cos(\theta_2 + \beta) & \cos(\theta_3 + \beta) & \cos(\theta_4 + \beta) \\ 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} u \quad (19)$$

We can then define the transformation matrix $\Omega_4(\beta)$ according to

$$\Omega_4(\beta) = \begin{bmatrix} \sin(\theta_1) & \sin(\theta_2) & \sin(\theta_3) & \sin(\theta_4) \\ \cos(\theta_1) & \cos(\theta_2) & \cos(\theta_3) & \cos(\theta_4) \\ 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}^{-1}$$
$$\begin{bmatrix} \sin(\theta_1+\beta) & \sin(\theta_2+\beta) & \sin(\theta_3+\beta) & \sin(\theta_4+\beta) \\ \cos(\theta_1+\beta) & \cos(\theta_2+\beta) & \cos(\theta_3+\beta) & \cos(\theta_4+\beta) \\ 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \tag{20}$$

3.3 Transformation matrix for $n = 5$

For the particular case of $n = 5$, shown in Fig. 8, an adequate choice for the complementary equations is shown in Eqs. 21.

$$\begin{matrix} \begin{bmatrix} 1 & -1 & 0 & 1 & -1 \end{bmatrix} \breve{u} = \begin{bmatrix} 1 & -1 & 0 & 1 & -1 \end{bmatrix} u \\ \begin{bmatrix} 1 & -1 & 0 & -1 & 1 \end{bmatrix} \breve{u} = \begin{bmatrix} 1 & -1 & 0 & -1 & 1 \end{bmatrix} u \end{matrix} \tag{21}$$
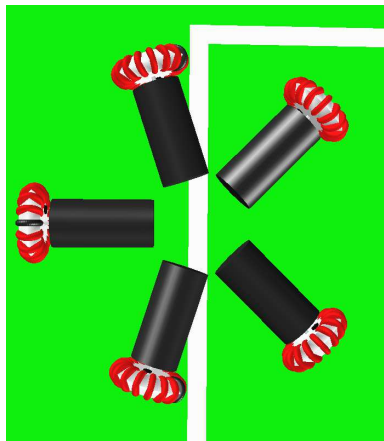

Fig. 8. Omnidirectional drive for $n = 5$

Equations 14, 15 and 21 can be put in matrix form as follows

$$\begin{bmatrix} \sin(\theta_1) & \sin(\theta_2) & \sin(\theta_3) & \sin(\theta_4) & \sin(\theta_5) \\ \cos(\theta_1) & \cos(\theta_2) & \cos(\theta_3) & \cos(\theta_4) & \cos(\theta_5) \\ 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 0 & 1 & -1 \\ 1 & -1 & 0 & -1 & 1 \end{bmatrix} \breve{u} =$$
$$\begin{bmatrix} \sin(\theta_1+\beta) & \sin(\theta_2+\beta) & \sin(\theta_3+\beta) & \sin(\theta_4+\beta) & \sin(\theta_5+\beta) \\ \cos(\theta_1+\beta) & \cos(\theta_2+\beta) & \cos(\theta_3+\beta) & \cos(\theta_4+\beta) & \cos(\theta_5+\beta) \\ 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 0 & 1 & -1 \\ 1 & -1 & 0 & -1 & 1 \end{bmatrix} u \tag{22}$$

We can then define the transformation matrix $\Omega_5(\beta)$ according to

$$
\Omega_5(\beta) = \begin{bmatrix} \sin(\theta_1) & \sin(\theta_2) & \sin(\theta_3) & \sin(\theta_4) & \sin(\theta_5) \\ \cos(\theta_1) & \cos(\theta_2) & \cos(\theta_3) & \cos(\theta_4) & \cos(\theta_5) \\ 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 0 & 1 & -1 \\ 1 & -1 & 0 & -1 & 1 \end{bmatrix}^{-1}
$$

$$
\begin{bmatrix} \sin(\theta_1+\beta) & \sin(\theta_2+\beta) & \sin(\theta_3+\beta) & \sin(\theta_4+\beta) & \sin(\theta_5+\beta) \\ \cos(\theta_1+\beta) & \cos(\theta_2+\beta) & \cos(\theta_3+\beta) & \cos(\theta_4+\beta) & \cos(\theta_5+\beta) \\ 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 0 & 1 & -1 \\ 1 & -1 & 0 & -1 & 1 \end{bmatrix} \tag{23}
$$

Further generalization is possible by taking into account that the set of $n-3$ complementary equations must be chosen such that the transformation matrix is full rank and of minimum condition number.

3.4 Global linearization using the transformation matrix $\Omega_n(\beta)$
The change of variable from u to $\breve{u}$ for any number of wheels n can now be expressed as

$$
\begin{aligned} \breve{u} &= \Omega_n(\beta)u \\ u &= \Omega_n(\beta)^{-1}\breve{u} \end{aligned} \tag{24}
$$

Matrix $\Omega_n(\beta)$ has the property of canceling the non-linear effect of β on matrix $B_1(\beta)$ since $B_1(0) = B_1(\beta)\Omega_n(\beta)^{-1}$ hence the model can be linearized at least with respect to the state variables in z_1, which are in fact the only variables we need to control. It is important to note that this is not a local linearization but a global one, so it is equally accurate in all of the control space.
By applying this change of variable to the state-space model of Eq. 13 we obtain the following model

$$
\begin{aligned}
\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} &= \begin{bmatrix} A_{11} & 0_{6\times n} \\ A_{21}(\beta) & 0_{n\times n} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} B_1(\beta) \\ 0_{n\times n} \end{bmatrix} \Omega_n(\beta)^{-1}\breve{u} \\
\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} &= \begin{bmatrix} A_{11} & 0_{6\times n} \\ A_{21}(\beta) & 0_{n\times n} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} B_1(\beta)\Omega_n(\beta)^{-1} \\ 0_{n\times n}\Omega_n(\beta)^{-1} \end{bmatrix} \breve{u} \\
\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} &= \begin{bmatrix} A_{11} & 0_{6\times n} \\ A_{21}(\beta) & 0_{n\times n} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} B_1(0) \\ 0_{n\times n} \end{bmatrix} \breve{u}
\end{aligned} \tag{25}
$$

If we separate this model according to the state vector partition proposed in Eq. 5 it is easy to see that the transformed model is linear with respect to state variables in z_1 and control variables $\breve{u}$

$$
\begin{aligned} \dot{z}_1 &= A_{11}z_1 + B_1(0)\breve{u} \\ \dot{z}_2 &= A_{21}(\beta)z_1 \end{aligned} \tag{26}
$$

From the state-space model in Eq. 26 it is possible to formulate a wide variety of linear state-space controllers for state variables in z_1.

4. Segment-wise optimal trajectory control

Our solution to the trajectory execution control problem requires the specification of the desired trajectory as a sequence of target regions as shown in Fig. 9. Each of the target regions is depicted as a red circle. This figure shows the robot in its initial state near the top left corner. The robot will have to visit each of the target regions in sequence until it reaches the final target which contains the red ball near the bottom right corner. As soon as the center of the robot enters the current target region it will start moving towards the next one. In this way, the trajectory is segmented by the target regions. The desired precision of the trajectory is determined by the size of the next target region. A small target requires higher precision than a larger one. Each target region makes additional specifications for the desired final state of the robot as it reaches the target. These specifications include the desired final scalar speed and the desired final heading direction. This last specification is useful for highly dynamic applications, such as playing soccer, since it allows the robot to rotate as it moves along the trajectory in order to align its manipulating device with the location of the target (a ball in the case of soccer).
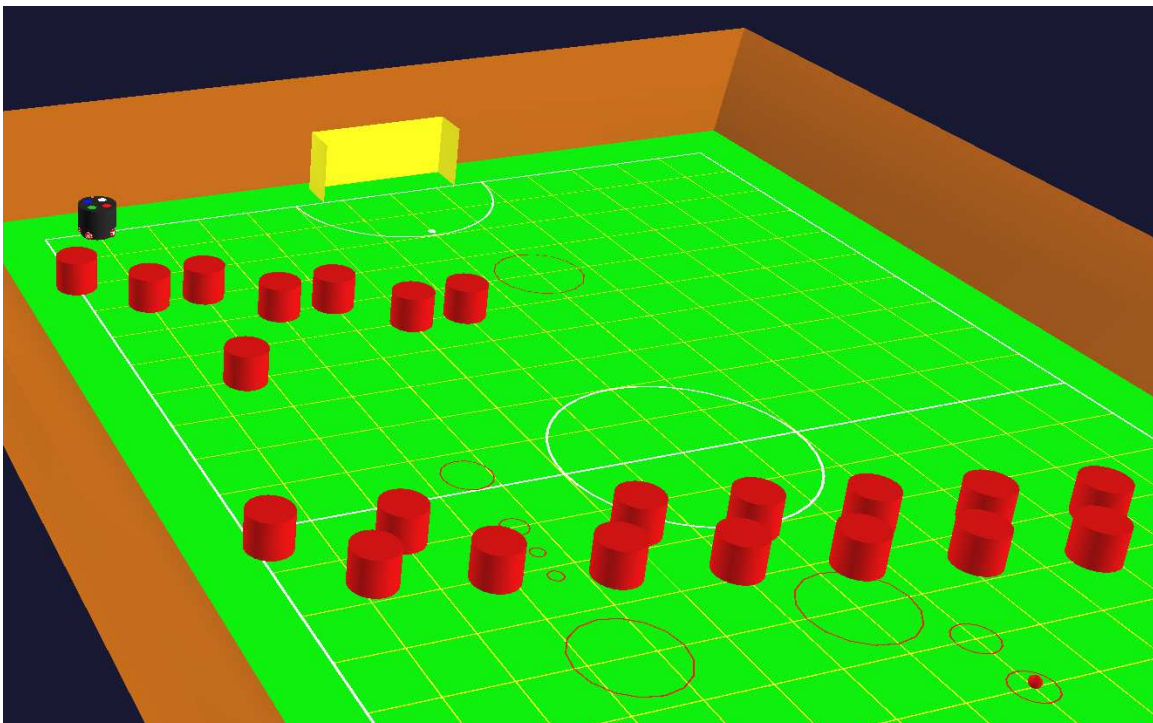

Fig. 9. Sequence of target regions

Each of the segments of the trajectory can be treated as a different control problem in which the final state of the previous segment is the initial state of the current one. Each segment has its own desired final state and in this way the mobile robot is forced to loosely follow the specified trajectory.

Once the trajectory has been segmented by this sequence of target regions the problem becomes how to force the robot to move from one target to the next. Our proposal is to solve an infinite-horizon LQR control problem for each of the segments. Although this approach may provide a sub-optimal solution rather than the optimal solution, that would be obtained from the corresponding finite-horizon LQR problem formulation, it has the advantage of requiring the solution of an algebraic Riccati equation rather than the more computationally demanding

differential Riccati equation. Computational efficiency is of course an important concern in this application since the solution will have to be obtained in real time in a high speed mobile robot environment.

The infinite-horizon LQR control problem consists on finding the state-feedback matrix K such that $\check{u} = -Kz_1$ minimizes the performance index J given by

$$J = \int_0^\infty (z_1{}^\mathrm{T} Q z_1 + \check{u}^\mathrm{T} R \check{u}) dt \tag{27}$$

taking into account that z_1 and $\check{u}$ are restricted by the dynamics of the mobile robot given by Eq. 26.

The performance index J specifies the total cost of the control strategy, which depends on an integral quadratic measure of the state z_1 and control $\check{u}$. Q and R represent positive definite matrices that give a weighted measure of the cost of each state variable and control variable, respectively.

For simplicity, in our solution Q and R are defined to be diagonal matrices

$$Q = \begin{bmatrix} w_{xy} & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{xy} & 0 & 0 & 0 & 0 \\ 0 & 0 & w_\beta & 0 & 0 & 0 \\ 0 & 0 & 0 & w_v & 0 & 0 \\ 0 & 0 & 0 & 0 & w_v & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{\dot\beta} \end{bmatrix} \tag{28}$$

$$R = w_m I_{n \times n}$$

where

- w_{xy} : cost weight of the XY position of the robot
- w_v : cost weight of the XY speed of the robot
- w_β : cost weight of the angular position of the robot
- $w_{\dot\beta}$: cost weight of the angular speed of the robot
- w_m : cost weight of the torque of the driving motors

This set of cost weights has to be specified for each of the segments of the trajectory. The weights specify the relative cost of each variable, and by an appropriate choice of their values one can easily adjust the optimality index for different control strategies. For example, if w_m is very large in comparison to the other weights then our strategy will be to save energy, if w_{xy} is large in comparison to the rest then the strategy dictates that the robot should reach the target region as soon as possible without regard to a specific target final speed, angle β or energy consumption. It would make sense to keep w_β very low during most of the trajectory except for those segments in which the robot is approaching the final target region or, in the case of soccer applications, orienting itself to receive a pass from a teammate robot.

The LQR approach is much more natural from the point of view of the designer as compared to other linear state-feedback controller design techniques such as *pole placement*. Although pole placement may help in specifying a desired time-domain transient behavior, the resulting feedback control may turn out to be higher than what the actuators can actually achieve. In LQR, on the other hand, the solution strikes a balance between the transient behavior of the state variables and the energy consumed by the actuators. In the case of LQR the resulting

poles are implicitly determined by the choice of matrices Q and R, rather than being explicitly specified.

Our segment-wise optimal solution to the trajectory execution control problem allows then to formulate control strategies that can easily adapt to the circumstances of the application. So, for example, the robot may be forced to move quickly in segments of the trajectory that do not require precision, and it may be left free to rotate in those segments in which its orientation is not of concern. In this way, its limited energy is consumed only for those objectives that are important at the moment.

5. Experimental results

A benchmark sequence of target regions was used for the purpose of testing our proposed solution. This sequence is shown in Fig. 9, and has a total of nine segments. Several obstacles were placed on the testing field to give more realism to the 3D OpenGL simulation of the resulting trajectory. A four wheel omnidirectional robot was used for this simulation with wheels distributed around its periphery according to Table 1. Wheels 1 and 4 are the frontal wheels. They are intentionally separated wider apart than the rest in order to allow for a manipulating device to be installed between them.

Table 1. Omnidirectional wheels distribution

θ_1	θ_2	θ_3	θ_4
$60°$	$135°$	$-135°$	$-60°$

The specific optimality-index parameters used for each of the segments of the trajectory are shown in Table 2. Throughout the whole trajectory, parameter w_m is kept at a moderate value in order to conserve energy without degrading the time-domain performance too much. Parameter w_β is kept at a low value for all segments except for the final two. This saves the energy that would otherwise be required to maintain the orientation of the robot towards a specific angle in the first seven segments where that would not be of interest, but then forces the robot to align itself with the location of the final target region in order to capture the object of interest located at the center of such region. Parameter $w_{\dot\beta}$ is kept at a low value throughout the whole trajectory. Parameter w_{xy} is set to a high value in segments 2, 3, 4, 5 and 9. The first four of these segments correspond to the part of the trajectory that takes the robot through the narrow opening in the obstacle barrier, where high precision is required in order to avoid a collision. High precision is also desirable in the last segment to ensure a successful capture of the object of interest. Finally, parameter w_v is given a moderate value throughout the first seven segments of the trajectory and then is lowered in the last two segments in order to allow for the energy to be used for the more important objective of orienting the robot towards the final target region. In this analysis what matters is the relative value of the weights. However, in order to give a more precise idea, in this analysis *moderate* means 1, *low* means 0.1 and *high* means 5.

The resulting trajectory is shown in Fig. 10. The continuous black line shows the path of the robot, while the smaller black straight lines show the orientation of the robot at each point. This gives an idea of how the robot rotates as it moves along the trajectory. One surprising result of this simulation experiment is that the robot started the second segment almost in reverse direction and slowly rotated to approach a forward heading motion. The final section of the trajectory shows how the object of interest gets captured by the robot. Figure 11 shows

Par.	1	2	3	4	5	6	7	8	9
w_{xy}	1	5	5	5	5	1	1	1	5
w_v	1	1	1	1	1	1	1	0.1	0.1
w_β	0.1	0.1	0.1	0.1	0.1	0.1	1	5	5
$w_{\dot\beta}$	1	0.1	0.1	0.1	0.1	1	0.1	0.1	0.1
w_m	1	1	1	1	1	1	1	1	1

Table 2. Optimality index parameters for each segment
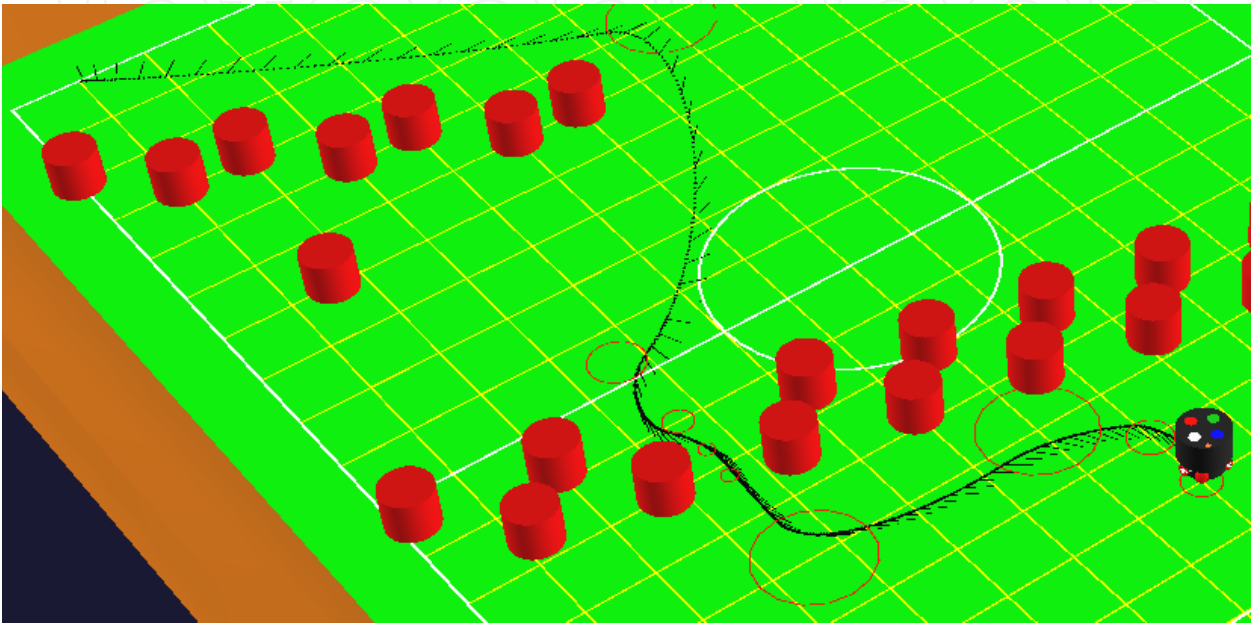
Fig. 10. Simulation of the segment-wise optimal trajectory execution

how the speeds and torques of each driving motor evolve against time. From these graphs we can interpret that there are only two required high energy transients. The first one comes at the initial time, when the robot must accelerate from a motionless state. The second high energy transient comes at the transition from the first to the second segment, where the robot must abruptly change direction while moving at a relatively high speed.
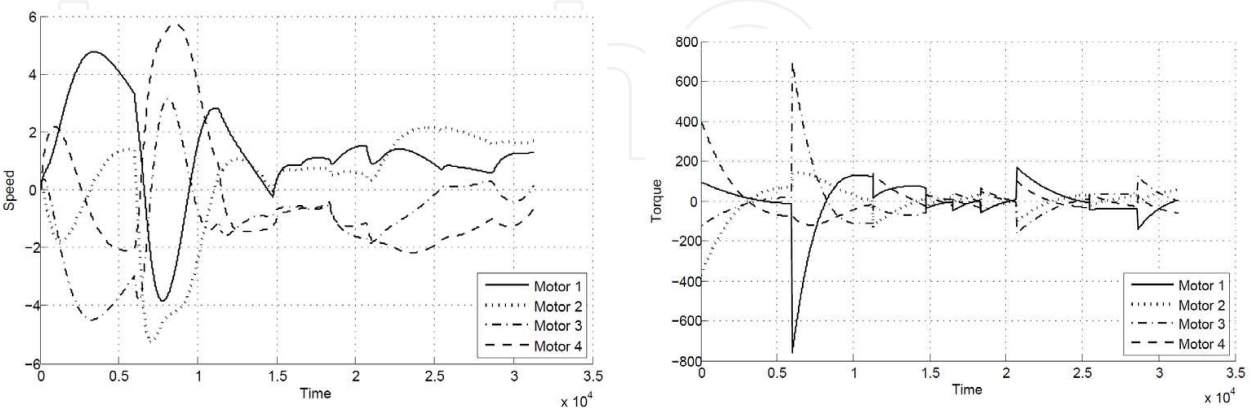

Fig. 11. Speed and torque vs. time for each wheel

A second experiment that we performed to test the segment-wise optimal solution to the trajectory execution problem is that of performing an 8-shaped trajectory, which was

successfully executed as shown in Fig. 12. Although the resulting 8 is not perfectly smooth the advantage here is that the path is achieved without the need for a computationally intensive path tracking algorithm but rather only loosely specified by a small set of target regions.
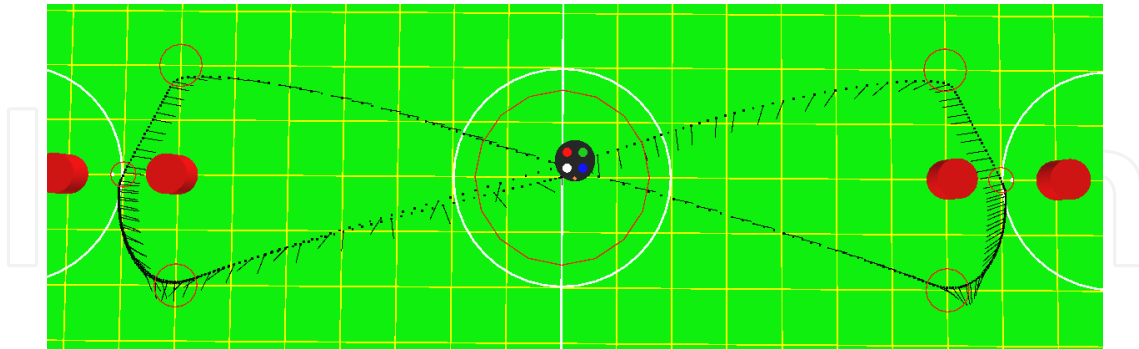

Fig. 12. Simulation of the segment-wise optimal 8-shaped trajectory

6. Conclusion

This work presents the formulation of a complete state-space model for the omnidirectional mobile robot. Although the model is nonlinear with respect to the control vector, it is shown how it can be globally linearized by a change of variables on the control vector. It has been shown how to apply this linearization for the case of $n = 3$, $n = 4$ and $n = 5$ omnidirectional wheels. The distinction among number of wheels is far from trivial since there is the need of introducing *complementary equations* that will make the proposed change of variable invertible.

The model used for analysis in this paper assumes an idealized omnidirectional robot with uniform density. Nonetheless, the same fundamental ideas that were tested under simulation in this work would still apply for a real robot.

There are still two major considerations to take into account before our proposed control method can be applied in the real world. First, our solution assumes availability of all state variables in z_1. However, in reality these state variables would have to be estimated from the available measurements using an approach similar to that presented in (Lupián & Avila, 2008). These measurements would come from the encoders of the driving motors and from the global vision system. The global vision system provides a measurement of variables x, y and β, but this measurement comes with a significant delay due to the computational overhead, so in actuality these variables would have to be predicted (Gloye et al., 2003). The second major consideration is that this solution will be implemented on a digital processor so, rather than a continuous-time model, the problem should be formulated in discrete-time.

The benchmark sequence of target regions used in the "Simulation results" section assumes the environment is static. In a real highly dynamic application the sequence of target regions will have to be updated as the trajectory execution progresses. This dynamic path-planning problem would be the task of a higher-level artificial intelligence algorithm which is left open for future research.

The first of the two simulation experiments can be better appreciated on the 3D OpenGL animation developed for this purpose. The reader can access a video of this animation online at (Lupián, 2009).

7. References

Ashmore, M. & Barnes, N. (2002). Omni-drive robot motion on curved paths: The fastest path between two points is not a straight-line, *AI '02: Proceedings of the 15th Australian Joint Conference on Artificial Intelligence*, Springer-Verlag, London, UK, pp. 225–236.

Balkcom, D. J., Kavathekar, P. A. & Mason, M. T. (2006a). The minimum-time trajectories for an omni-directional vehicle, *Seventh International Workshop on the Algorithmic Foundations of Robotics*, New York City.

Balkcom, D. J., Kavathekar, P. A. & Mason, M. T. (2006b). Time-optimal trajectories for an omni-directional vehicle, *Int. J. Rob. Res.* 25(10): 985–999.

Borenstein, J. & Evans, J. (1997). The omnimate mobile robot: Design, implementation, and experimental results, *International Conference on Robotics and Automation, IEEE*, Albuquerque, NM, pp. 3505–3510.

Gloye, A. & Rojas, R. (2006). *Künstliche Intelligenz*, Springer, chapter Holonomic Control of a Robot with an Omnidirectional Drive.

Gloye, A., Simon, M., Egorova, A., Wiesel, F., Tenchio, O., Schreiber, M., Behnke, S. & Rojas, R. (2003). Predicting away robot control latency, *Technical report B-08-03*, FU-Berlin.

Kalmár-Nagy, T., D'Andrea, R. & Ganguly, P. (2004). Near-optimal dynamic trajectory generation and control of an omnidirectional vehicle, *Robotics and Autonomous Systems* 46: 47–64.

Kalmár-Nagy, T., Ganguly, P. & D'Andrea, R. (2002). Real-time trajectory generation for omnidirectional vehicles, *American Control Conference*, Anchorage, AK, pp. 286–291.

Lupián, L. F. (2009). Simulación de robot F180 en AIR-GL, World Wide Web electronic publication.
    URL: *http://www.youtube.com/watch?v=-rBP3VwDPAs*

Lupián, L. F. & Avila, R. (2008). Stabilization of a wheeled inverted pendulum by a continuous-time infinite-horizon LQG optimal controller, *Robotic Symposium, IEEE Latin American*, Bahia, Brazil, pp. 65–70.

Lupián, L. F. & Rabadán-Martin, J. R. (2009). Segment-wise optimal trajectory execution control for four-wheeled omnidirectional mobile robots, *Robotics Symposium (LARS), 2009 6th Latin American*, Valparaíso, Chile, pp. 1 – 6.

Purwin, O. & D'Andrea, R. (2005). Trajectory generation for four wheeled omnidirectional vehicles, *American Control Conference* pp. 4979–4984.
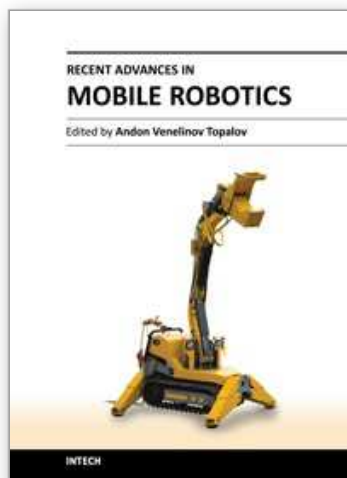
Recent Advances in Mobile Robotics

Edited by Dr. Andon Topalov

Mobile robots are the focus of a great deal of current research in robotics. Mobile robotics is a young, multidisciplinary field involving knowledge from many areas, including electrical, electronic and mechanical engineering, computer, cognitive and social sciences. Being engaged in the design of automated systems, it lies at the intersection of artificial intelligence, computational vision, and robotics. Thanks to the numerous researchers sharing their goals, visions and results within the community, mobile robotics is becoming a very rich and stimulating area. The book Recent Advances in Mobile Robotics addresses the topic by integrating contributions from many researchers around the globe. It emphasizes the computational methods of programming mobile robots, rather than the methods of constructing the hardware. Its content reflects different complementary aspects of theory and practice, which have recently taken place. We believe that it will serve as a valuable handbook to those who work in research and development of mobile robots.

How to reference

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

Luis F. Lupián and Josué R. Rabadán-Martin (2011). LQR Control Methods for Trajectory Execution in Omnidirectional Mobile Robots, Recent Advances in Mobile Robotics, Dr. Andon Topalov (Ed.), ISBN: 978-953-307-909-7, InTech, Available from: http://www.intechopen.com/books/recent-advances-in-mobile-robotics/lqr-control-methods-for-trajectory-execution-in-omnidirectional-mobile-robots

INTECH
open science | open minds