Intermediate Project
Dynamic control of mobile robot with differential drive

Author: Hubert Grzegorczyk

Supervised by: Dr inż. Witold Paluszynski
Department of Electronics
Wrocław University of Science and Technolology

January 30, 2019

Abstract:
The goal of this project was to create dynamic controller for mobile robot with differential drive. The dynamic control algorithm should be reliable and simple. Trajectory generation should be part of the research. Algorithm will be also implemented on the embedded driver of a mobile robot.

1 Introduction

The goal of this project was to create a reliable dynamic control algorithm for the mobile robot (differential drive). The robot is ready, previously developed mobile platform and it has basic algorithms for the kinematic control. Dynamic control algorithms should allow better control of the robot, specially at a higher speeds of the robot (over 1 m/s). Concept of the problem was taken from the *Micromouse* robotic competition in which robot has to travel from the start to the end of the maze in the shortest possible time. The algorithm should be developed on PC in simulation and later implemented on embedded robot driver. There are possible constraints on this project like computing power of embedded robot driver or difficulty of motors torque control in a simple mobile robot.

2 Project

2.1 Dynamic robot model

The dynamic differential drive robot model was prepared based on [1]:

$$\overline{M}(q)\ddot{\eta} + \overline{V}(q,\dot{q})\dot{\eta} = \overline{B}(q)\tau \tag{1}$$

$$\begin{aligned}
\overline{M}(q) &= \begin{bmatrix} I_w + \frac{R^2}{4L^2}(mL^2+I) & \frac{R^2}{4L^2}(mL^2-I) \\ \frac{R^2}{4L^2}(mL^2-I) & I_w + \frac{R^2}{4L^2}(mL^2+I) \end{bmatrix} \\
\overline{V}(q,\dot{q}) &= \begin{bmatrix} 0 & \frac{R^2}{2L}m_c d\dot{\theta} \\ -\frac{R^2}{2L}m_c d\dot{\theta} & 0 \end{bmatrix} \\
\overline{B}(q) &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
\end{aligned} \tag{2}$$

where:

- q is state vector,
- η is wheels position,
- τ is torque on motors,
- I_w is wheel inertia,
- R is is wheel radius,
- L is wheel distance from robot centre,
- m is mass of a robot,
- I is total equivalent inertia,
- m_c is robot main rigid body mass,
- d is position of mass centre in x axis of robot frame of the reference.

2.2 Dynamic control algorithm

Simple dynamic controller can be prepared using dynamic model of the robot. Equation (1) can be divided by $\overline{B}$. This allows to simply calculate torque needed on motors to achieve desired acceleration ($\ddot{\eta}$).

$$\tau = \overline{B}^{-1}(q)(\overline{M}(q)\ddot{\eta} + \overline{V}(q,\dot{q})\dot{\eta}) \tag{3}$$

Speeds needed to follow the trajectory are obtained from Samson controller which guarantees convergence to the desired trajectory.

2.3 Bézier curve trajectory generator

Trajectories are generated with use cubic of Bézier curve [2]. Example curve is shown on figure below:

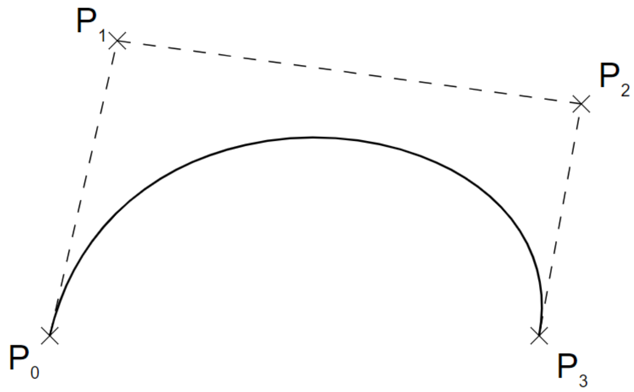

Figure 1: Example cubic Bézier curve

This curve is described by four points P_0, P_1, P_2, P_3. Points on curve are described by equation:

$$P(t) = \begin{bmatrix} P_x(t) \\ P_y(t) \end{bmatrix} = \begin{bmatrix} P_0 & P_1 & P_2 & P_3 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}. \tag{4}$$

Where $t \in < 0, 1 >$.
Also curvature along the curve can be calculated:

$$K = \frac{P'_x P''_y - P'_y P''_x}{(P'^2_x + P'^2_y)^{\frac{3}{2}}}. \tag{5}$$

In practical use needed is arc length parametrization of the curve P(s) (position on path depending on how far robot travelled along path) and curvature K(s). This can be approximated numerically. For example 1000 points can be calculated along the path and sum of distance between those points can be used as estimation of arc length.

Cubic bézier curve trajectory generator was implemented on robot driver. In this case robot at the start is always in point P_0 (centre of robot frame of reference) directed towards P_1. Position of P_1, P_2, P_3 describes trajectory of a robot.

2.4 Simulation

During development of the algorithms simulation in Matlab was used. Dynamic 2 and kinematic 3 controllers were prepared. Kinematic controller uses PID to control torque on wheels to obtain desired speeds. Physical parameters in dynamic model were measured on the real robot.

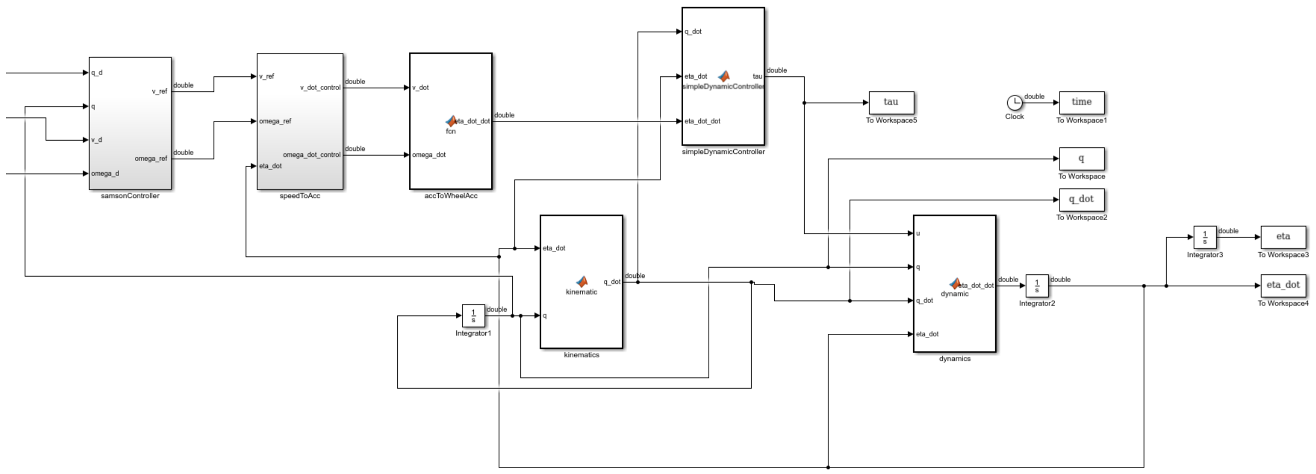

Figure 2: Overall view of the dynamic control algorithm in Simulink

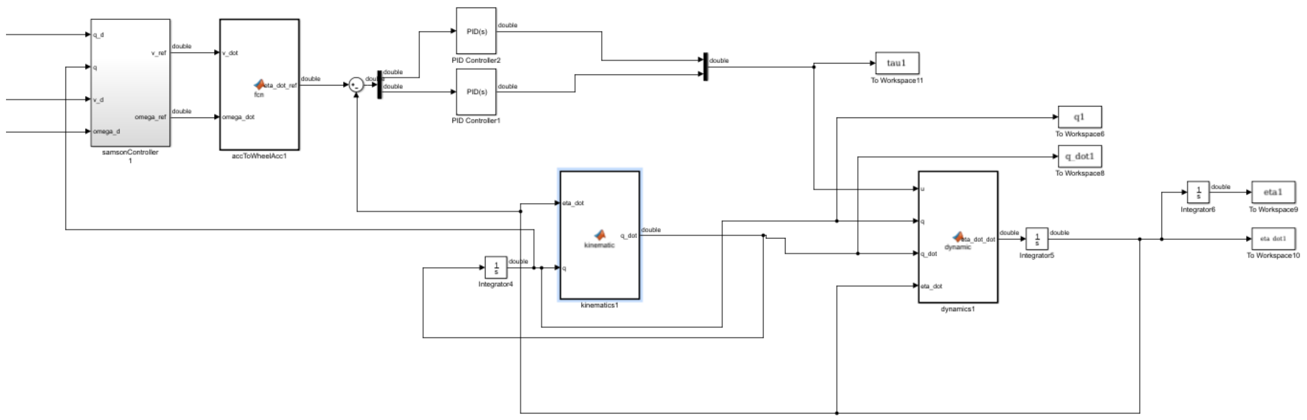

Figure 3: Overall view of the kinematic control algorithm in Simulink

2.5 Resource usage analysis

Resource usage of this algorithms is not high. Dynamic controller just uses simple calculations. Only problem was with Bézier curves trajectory generator. To calculate curvature of curve and position in global frame of reference in arc length parametrization one need to iterate over some number of points on curve. This was problem on embedded driver of the robot. This calculation took too long and to make it faster that points were stored in memory. Even then execution of the trajectory generator on the robot was too long and its frequency was changed from 500Hz to 200Hz.

3 Results

Simple dynamic controller was prepared. Trajectory generator based on Bézier curves was implemented in simulation and also on the embedded driver of the robot.

3.1 Comparison between dynamic and kinematic controller

To check utility of dynamic controller it was compared with kinematic controller in the simulation. Results of the simulation are shown on figure 4. Robot started with small disturbance, it was not on the path.

As it can be seen on graph dynamic controller is working better in path following task.

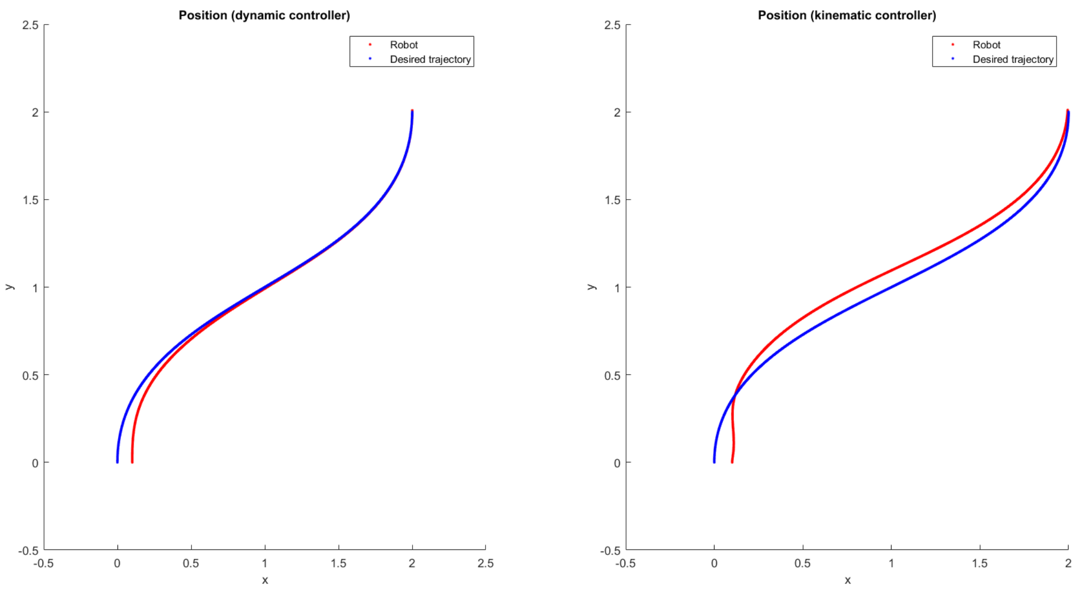

Figure 4: Simulation results for example trajectory

4 Conclusion

Part of the main goals was completed. Implemented algorithm is working well in the simulation. Trajectory generator can produce nice, continuous and smooth trajectories both in simulation and on the robot's driver. Robot is able to travel from one point to another along given cubic Bézier curve. Implementation of dynamic controller on embedded robot's driver can cause some problems, torque control requires special motor drivers and motors with known characteristics.

5 Software and licenses

- Matlab & Simulink toolbox — 30-day trial license
- STM32 Workbench — free tool

References

[1] Rached Dhaouadi and Ahmad Abu Hatab [*Dynamic Modelling of Differential-Drive Mobile Robots using Lagrange and Newton-Euler Methodologies: A Unified Framework*].
Link

[2] Wikipedia about Bézier curves
`https://en.wikipedia.org/wiki/Bézier_curve`