https://doi.org/10.47734/regrasp.v10.03.p32-44

CONTROL BARRIER FUNCTIONS APPLIED TO AN OMNIDIRECTIONAL MOBILE ROBOT

FUNÇÕES DE BARREIRA DE CONTROLE APLICADAS À UM ROBÔ MÓVEL OMNIDIRECIONAL

Félix Gabriel dos Santos Souza
Undergraduate Student in Control and Automation Engineering
IFSP/Campus São Paulo
felix.santos@aluno.ifsp.edu.br

Caio Igor Gonçalves Chinelato
Ph.D. in Electrical Engineering /POLI-USP
Professor and Researcher at CECS (Center for Engineering, Modeling and Applied Social Sciences)/UFABC
caio.chinelato@ufabc.edu.br; caio.chinelato@gmail.com

ARTIGO INFO. Recebido: 16.04.2025 Aprovado: 15.08.2025 Disponibilizado: 01.09.2025

ABSTRACT

Control barrier functions (CBFs) have been extensively studied recently and are related to the safety of control systems. Safety is represented by constraints on the states and outputs of the system. The final control framework integrates the stabilization/tracking objectives, described by a control Lyapunov function (CLF) or a nominal control input, and the safety constraints, described by CBFs, through a quadratic programming (QP). The objective and the main contribution of this work is the application of this control framework to the Robotino mobile robot (Festo), which is used in several educational institutions in Brazil and worldwide, and unlike many didactic mobile robots, it features a more robust structure, higher actuation power, and is omnidirectional, making it more compatible with real-world applications. Several studies in the literature propose controlling the Robotino to satisfy stabilization/tracking objectives; however, none of these studies apply CBFs to ensure that safety constraints are respected. The results, obtained through computational simulations, demonstrate that both the stabilization/tracking objectives, represented by reference trajectories, and the safety constraints, represented by position and coordinate constraints of the robotic system, were satisfied.

Keywords: Control Barrier Function; Safety; Robotic Systems; Omnidirectional Mobile Robot; Control Lyapunov Function.

RESUMO

Funções de barreira de controle (FBCs) têm sido bastante estudadas recentemente e estão relacionada com segurança de sistemas de controle. A segurança é representada por restrições nos estados e nas saídas do sistema. A estrutura de controle final integra os objetivos de estabilização/rastreamento, descritos por uma função de *Lyapunov* de controle (FLC) ou por uma entrada de controle nominal, e as restrições de segurança, descritas por FBCs, por meio de uma programação quadrática (PQ). O objetivo e a principal contribuição deste trabalho é a aplicação desta estrutura de controle no robô móvel *Robotino* (*Festo*), que é usado em diversas instituições de ensino no Brasil e o no mundo, e diferentemente de muitos robôs móveis didáticos, apresenta uma estrutura mais robusta, maior potência de acionamento e é omnidirecional, sendo mais compatível com aplicações práticas reais. Diversos trabalhos apresentados na literatura propõem o controle do *Robotino* para satisfazer objetivos de estabilização/rastreamento, no entanto, nenhum destes trabalhos aplicam FBCs para que restrições de segurança sejam respeitadas. Os resultados, obtidos através de simulações computacionais, demonstram que os objetivos de estabilização/rastreamento, representados por trajetórias de referência, e as restrições de segurança, representadas por restrições de posições e de coordenadas do sistema robótico, foram satisfeitos.

Palavras-chave: Função de Barreira de Controle; Segurança; Sistemas Robóticos; Robô Móvel Omnidirecional; Função de *Lyapunov* de Controle.

Introduction

Safety is a fundamental concept in several control engineering problems, such as robotic systems and automotive applications, for example. Systems considered safe are those that must satisfy traditional control objectives, such as stabilization/tracking, and safety constraints, where the safety constraints must be prioritized. Recently, the study of safe control systems has received considerable attention in the field of control theory. Safety is represented by constraints on the states and outputs of the system. These constraints are specified by an invariant set, defined by a control barrier function (CBF).

Considering a dynamical system

$$\dot{x} = f(x) + g(x)u, \quad (1)$$

where $x \in \mathbb{R}^n$ are the states and $u \in \mathbb{R}^m$are the inputs, and a safe set *C* related to the system safety, we have that *h(x)*$\rightarrow$*0* as *x*$\rightarrow \partial$*C*, where *h(x)* is denominated CBF. If *h(x)* satisfies specific conditions, then the invariance of *C* is guaranteed and the system is safe (Ames et al., 2019).

A controller must be designed for the system considered in Equation (1) with the objective of keeping the system states within the safe set *C*, defined as (Ames et al., 2017):

$$\begin{gathered} C = \{x \in \mathbb{R}^n : h(x) \geq 0\}, \\ \partial C = \{x \in \mathbb{R}^n : h(x) = 0\}, \\ int(C) = \{x \in \mathbb{R}^n : h(x) > 0\}. \quad (2) \end{gathered}$$

Considering the safe set *C*, the CBF *h(x),* and the set

$$K_{cbf}(x) = \{u \in U : L_f h(x) + L_g h(x)u + \alpha(h(x)) \geq 0\}, \quad (3)$$

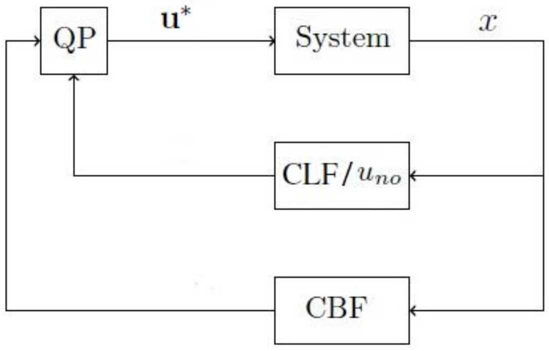
we have that any controller *u(x)* $\in$ $K_{cbf}(x)$ will render the set *C* invariant and the system safe (Ames et al., 2017). The terms $L_f h(x) = \nabla h(x).f(x)$ and $L_g h(x) = \nabla h(x).g(x)$ are related to the Lie derivatives.

The final control framework integrates the stabilization/tracking objectives, described by a control Lyapunov function (CLF) or a nominal control input u_{no}, and the safety constraints, described by CBFs, through a quadratic programming (QP), which is an optimization problem that aims to minimize a quadratic cost or objective function subject to equality or inequality constraints, and can be expressed in the form

$$\begin{aligned} & argmin_z \frac{1}{2} z^T E z + q^T z \\ & s.t \; Gz \leq W \\ & \quad G_{eq} z = W_{eq}, \quad (4) \end{aligned}$$

where *E*, *G* and G_{eq} are matrices, *q*, *W* and W_{eq} are vectors, and *z* is the variable to be minimized (Borrelli et al., 2017). The control framework ensures that the safety constraints take priority over the stabilization/tracking objectives. Figure 1 presents a synthesized description of the control framework.

Figure 1. Synthesized description of the control framework.



Source: Authors.

When the stabilization/tracking objectives are described by a CLF *V(x)*, the expression for the final controller is given by (Ames et al., 2017):

$$\begin{aligned} \boldsymbol{u}^*(x) = \; & argmin_{\boldsymbol{u}=(u,\delta)\in\mathbb{R}^m\times\mathbb{R}}\left[\frac{1}{2}\boldsymbol{u}^T H(x)\boldsymbol{u} + F(x)^T\boldsymbol{u}\right] \\ s.t. \quad & L_f V(x) + L_g V(x)u + cV(x) - \delta \le 0 \\ & L_f h(x) + L_g h(x)u + \alpha(h(x)) \ge 0, \quad (5) \end{aligned}$$

where *H(x)* and *F(x)* are matrices related to the system, and *δ* is the relaxation parameter used to prioritize the safety constraints. Typically *α(h(x)) = γh(x)*, where *γ* is a positive constant and a project parameter related to the CBF, and *c* is a project parameter related to the CLF.
When the stabilization/tracking objectives are described by a nominal control input u_{no}, the expression for the final controller is given by (Rauscher et al., 2016), (Gurriet et al., 2018), (Ames et al., 2019):

$$\begin{aligned} \boldsymbol{u}^*(x) = \; & argmin_{\boldsymbol{u}\in\mathbb{R}^m}[u^T u - 2{u_{no}}^T u] \\ s.t. \quad & L_f h(x) + L_g h(x)u + \alpha(h(x)) \ge 0. \quad (6) \end{aligned}$$

This control framework is also known as a safety filter, as it filters the nominal control input u_{no} to satisfy the safety constraints imposed by the CBF. It is important to highlight that u_{no} is typically represented by a linear or nonlinear control law. In some cases, as shown in Igarashi and Nakamura (2018), this input can be provided by an human operator, for example.
Here it is important to highlight that all the formulation of the control framework described so far has been based on the zeroing CBF *h(x)*. However, we can also consider the reciprocal CBF *B(x)*. Some studies adopt *h(x)* while others adopt *B(x)*, nonetheless, the results are very similar. The difference between the two approaches lies in the fact that, considering the set *C* defined in Equation (2), which guarantees system safety, we have that $B(x)\to\infty$ as $x\to\partial C$ and $h(x)\to 0$ as $x\to\partial C$. Typically, *B(x) = 1/h(x)* is adopted, and the CBF constraints in Equations (5) e (6) are now given by (Ames; Grizzle; Tabuada, 2017):

$$L_f B(x) + L_g B(x)u - \alpha(B(x)) \le 0. \quad (7)$$

The safe control framework with CBF, shown in Figure 1 and in Equations (5) and (6), was initially presented by Ames et al. (2014), and several applications using this methodology are reported in the literature. In Mehra et al. (2015), this control framework is applied to the adaptive cruise control problem, where an autonomously controlled vehicle must reduce or adapt its speed to maintain a safe distance from other vehicles on the road, while also ensuring vehicle's passengers comfort. Still in the context of autonomous vehicles, Xu et al. (2017) employ the control framework to keep the vehicle within its lane. Gurriet et al. (2018) demonstrate the application of the control framework to a Segway, a two-wheeled transporter, where the controller's CBFs act to prevent the user from falling. More recent studies have also applied this control framework to robotic systems, such as robotic manipulators (Ducaju et al., 2022; Ferraguti et al., 2020), quadcopters (Singletary et al., 2022; Yang et al., 2022), and bipedal robots (Peng et al., 2023; Ahmadi et al., 2022).
The objective and the main contribution of this work is the application of this control framework to the Robotino mobile robot, manufactured by Festo (Robotino, 2025). Robotino is a wheeled mobile robot used in several educational institutions in Brazil and worldwide, and unlike many didactic wheeled mobile robots, such as TurtleBot3 (Turtlebot3, 2025) and Pololu 3pi+ 2040 (Pololu, 2025), it features a more robust structure, higher actuation power, and is omnidirectional (Siegwart et al., 2011), making it more compatible with real-world applications. Furthermore, Festo provides a dedicated Robotino simulator and libraries that allow the robot to be programmed in several ways, such as Matlab, C/C++, Java, etc (Robotino, 2025). Therefore, simulations that approximate a real-world scenario can be developed more easily, and the robot can be programmed in multiple ways.

Several studies in the literature propose controlling Robotino to satisfy stabilization/tracking objectives, such as El-Sayyah et al. (2025) and Hedman and Mercorelli (2021); however, none of these studies apply CBFs to ensure that safety constraints, such as obstacle avoidance for example, are respected. Some studies apply CBFs to mobile robots with smaller wheels and differential traction (non-omnidirectional), for instance, Sa et al. (2024), Manjunath and Nguyen (2021), and Toulkani et al. (2022).
The results, obtained through computational simulations, demonstrate that both the stabilization/tracking objectives, represented by reference trajectories, and the safety constraints, represented by position and coordinate constraints of the robotic system, were satisfied. The following sections present the activities realized for the development of this work, the results, the conclusions, and the future works.

Methodology

In order to apply the control framework with CBF, described in Figure 1 and Equations (5) and (6), the robotic system must be represented in the form of Equation (1). Therefore, the first step of the methodology was to establish the mathematical modeling of the Robotino mobile robot. The subsequent step involved the study and determination of the CLF and the nominal control input, to ensure that the stabilization/tracking objectives are satisfied, as well as the CBF, to guarantee that the safety constraints are satisfied.
Posteriorly, computational simulations were carried out to evaluate the performance of the control framework applied to the mobile robot. The initial simulation results were obtained by implementing the control framework in Matlab, considering only the Robotino mathematical model, also simulated in Matlab. The objective of these initial simulations was to understand the operation of the control framework and adjust the controller parameters, since, under this more idealized condition, simulations can be performed more quickly and directly. Subsequently, results were obtained using the RobotinoSim simulator integrated with Matlab. RobotinoSim is the dedicated Robotino simulator and provides a complete simulation environment including the robot, sensor readings, obstacles, etc. The control framework was again implemented in Matlab, however, the Robotino was simulated within the RobotinoSim environment. For this, the Robotino library for Matlab was used, which includes blocks for inputs of wheel angular velocities, odometry, infrared distance sensors, camera, among others. The objective of these final simulations was to verify the performance of the control framework in a scenario closer to reality using the robotic system simulator. In both cases, the simulation scenario considered the robot following a reference trajectory from an initial position to a final position (stabilization/tracking objective). An obstacle was placed between the initial and final positions, and the robot was required to avoid the obstacle through the application of the CBF (safety constraint).

Robotino Mobile Robot – Description and Mathematical Modeling

The Robotino mobile robot, manufactured by Festo and shown in Figure 2 (Robotino, 2025), features three independent omnidirectional wheels, nine infrared distance sensors, incremental encoders with Proportional-Integral-Derivative (PID) control in the drive units, a collision detection sensor (bumper) mounted around its circumference, and an USB interface camera. It is important to highlight that, although the results were obtained through computational simulations, understanding and studying the structure, operation, and sensors of the robotic system is essential, since the Robotino simulator replicates a scenario very close to reality.

Figure 2. Robotino omnidirectional mobile robot.

Source: Robotino, (2025).

As previously discussed, in order to apply the control framework analyzed in this work, the robotic system must be represented in the form of Equation (1). Therefore, the mathematical modeling of the Robotino mobile robot will be shown below.

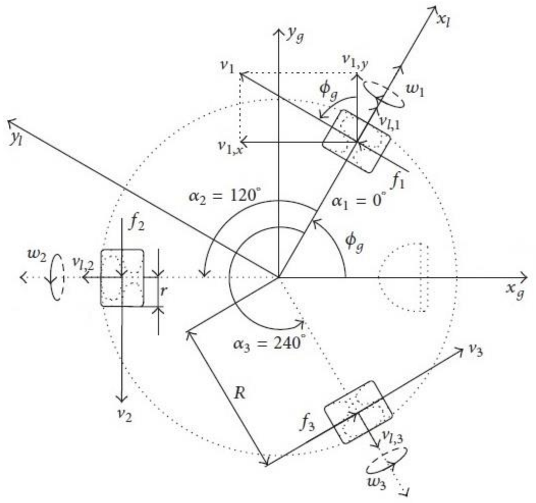
Figure 3 presents the geometric and kinematic relations of Robotino (Tang & Eberhard, 2013). The three omnidirectional wheels are separated by 120°, and the robot coordinate system (x_l, y_l) and the global coordinate system (x_g, y_g) are shown, along with the rotation angle ϕ_g, the translational velocities v_1, v_2 and v_3, the lateral velocities $v_{l,1}$, $v_{l,2}$ and $v_{l,3}$, and the actuation forces f_1, f_2 and f_3. To describe the position of the three wheels with respect to the robot coordinate system, the angles $\alpha_1 = 0°$, $\alpha_2 = 120°$ and $\alpha_3 = 240°$ are used.

The relation between the global velocities of the robot and the translational velocities of the three wheels is given by (Tang & Eberhard, 2013):

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} -\sin(\phi_g + \alpha_1) & \cos(\phi_g + \alpha_1) & R \\ -\sin(\phi_g + \alpha_2) & \cos(\phi_g + \alpha_2) & R \\ -\sin(\phi_g + \alpha_3) & \cos(\phi_g + \alpha_3) & R \end{bmatrix} \begin{bmatrix} \dot{x}_g \\ \dot{y}_g \\ \dot{\phi}_g \end{bmatrix}, \quad (8)$$

where R is the distance from the center of the robot to the center of the wheel.

Figure 3. Geometric and kinematic relations of Robotino.



Source: Tang & Eberhard, (2013).

Considering the relation $v_i = r\omega_i$, where ω_i (i = 1, 2, 3) are the angular velocities of the wheels with radius r, and rearranging (8) into a constant part, which depends on the parameters α_1, α_2 and α_3, and a variable part, we obtain (Tang & Eberhard, 2013):

$$\begin{bmatrix}\omega_1\\\omega_2\\\omega_3\end{bmatrix} = S_g \begin{bmatrix}\dot{x}_g\\\dot{y}_g\\\dot{\phi}_g\end{bmatrix}, \quad (9)$$

such that

$$S_g = \frac{1}{r}\begin{bmatrix}0 & 1 & R\\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & R\\ \frac{\sqrt{3}}{2} & -\frac{1}{2} & R\end{bmatrix}\begin{bmatrix}\cos(\phi_g) & \sin(\phi_g) & 0\\ -\sin(\phi_g) & \cos(\phi_g) & 0\\ 0 & 0 & 1\end{bmatrix}. \quad (10)$$

For the application of the control framework analyzed in this work, the system must be represented in the form of Equation (1). We consider the control inputs $u = [\omega_1\ \omega_2\ \omega_3]^T$, states $x = \left[x_g\ y_g\ \phi_g\right]^T$, *f(x)* = 0, and *g(x)* = S_g^{-1}. Therefore, the final model of the system is given by:

$$\begin{bmatrix}\dot{x}_g\\\dot{y}_g\\\dot{\phi}_g\end{bmatrix} = S_g^{-1}\begin{bmatrix}\omega_1\\\omega_2\\\omega_3\end{bmatrix}. \quad (11)$$

Application of the Control Framework with CBF

The control framework with CBF, shown in Figure 1 and Equations (5) and (6), was applied so that the robotic system satisfies stabilization/tracking objectives, represented by reference trajectories, and safety constraints, represented by position and coordinate constraints, ensuring that the robotic system avoids obstacles.

In order to satisfy the stabilization/tracking objectives, the following CLF was used (Elsayed et al., 2017):

$$V = \frac{(x_d - x)^2 + (y_d - y)^2 + (\phi_d - \phi)^2}{2}, \quad (12)$$

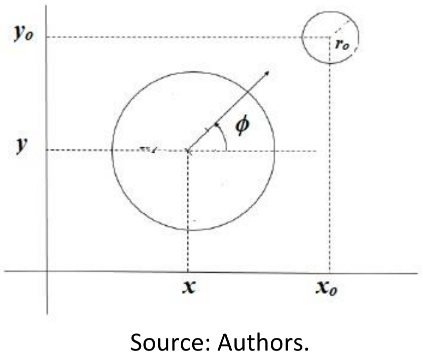
where (*x, y, ϕ*) are the current robot coordinates values and (x_d, y_d, ϕ_d) are the desired robot coordinates values given by the reference trajectory.

In order to satisfy the safety constraints, the following CBF was used (Igarashi; Nakamura, 2018):

$$B = \sum_{o=1}^{n} \frac{1}{(x - x_o)^2 + (y - y_o)^2 - (r_o + R + r_c)^2} + x^2 + y^2, \quad (13)$$

where (x_o, y_o) are the coordinates of each obstacle, r_o is the radius of each detected obstacle, as shown in Figure 4, and r_c is the distance from the center of the wheel to robot's edge. Here it is important to highlight that the reciprocal CBF *B(x)* is applied.

Figure 4. Schematic representation of the robot and the obstacle.



Source: Authors.

For the analyzed problem, the control framework in Equation (5) is given by:

$$\boldsymbol{u}^*(x) = \ argmin_{\boldsymbol{u}=(\omega_1,\omega_2,\omega_3,\delta)^T \in \mathbb{R}^3 \times \mathbb{R}} \left[\frac{1}{2}\boldsymbol{u}^T H(x)\boldsymbol{u}\right]$$

$$\begin{aligned} s.t. \quad & A_{clf}\boldsymbol{u} \leq b_{clf} \\ & A_{cbf}\boldsymbol{u} \leq b_{cbf}, \end{aligned} \quad (14)$$

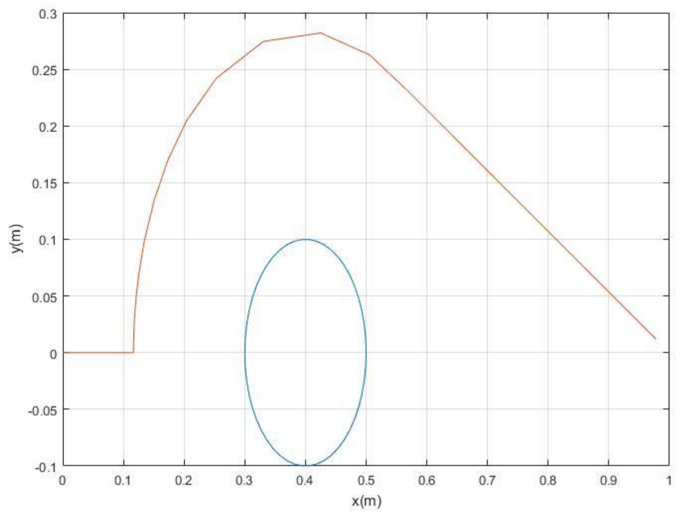
Where $H(x) = diag(p_1, \ldots, p_4)$, $A_{clf} = [L_gV(x), -1]$, $b_{clf} = -L_fV(x) - cV(x)$, $A_{cbf} = [L_gB(x), 0]$, and $b_{cbf} = -L_fB(x) + \frac{\gamma}{B(x)}$. The matrix *H(x)* is a matrix whose weights $p_i > 0$ are related to each control input, δ is the relaxation parameter used to prioritize the safety constraints, *c* is the CLF design parameter related to the convergence rate for stabilization when the CLF is acting to satisfy the stabilization/tracking objectives, and *γ* is the CBF design parameter that specifies how far from the barrier limit the CBF acts to satisfy the safety constraints.

Results and Discussions

Finally, to verify whether the control framework in Equation (14) satisfies the stabilization/tracking objectives and the safety constraints, computational simulations were conducted. As presented in the Methodology section, the initial simulation results were obtained by implementing the control framework in Matlab, considering only the Robotino mathematical model, also simulated in Matlab, aiming to understand and adjust the controller parameters in a more idealized condition. Subsequently, results were obtained using the RobotinoSim simulator integrated with Matlab, which is the dedicated Robotino simulator and provides a complete simulation environment including the robot, sensor readings, obstacles, etc. The control framework was again implemented in Matlab, however, the Robotino was simulated within the RobotinoSim environment using the Robotino library for Matlab, which includes blocks for inputs of wheel angular velocities, odometry, infrared distance sensors, camera, among others. With these final simulations, the performance of the control framework can be evaluated in a scenario closer to reality using the considered robotic system simulator.

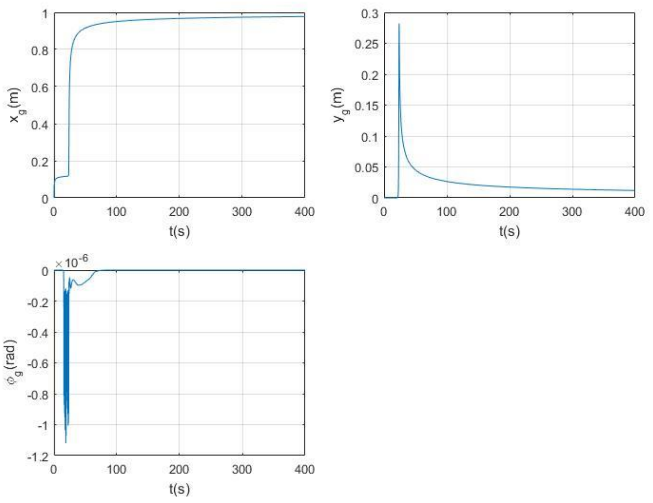
A simulation scenario was considered in Matlab where the robot must follow a reference trajectory starting from the initial position (x_i, y_i, ϕ_i) = (0 m, 0 m, 0 rad) and arriving at the final position (x_f, y_f, ϕ_f) = (1 m, 0 m, 0 rad). An obstacle is placed between the initial and final positions, and the robot must avoid it. The obstacle has coordinates (x_o, y_o) = (0.4 m, 0 m) and radius r_o = 0.1 m. The parameters considered were *r* = 0.04 m, *R* = 0.125 m, *c* = 1000, *γ* = 1000, p_1 = 1, p_2 = 1, p_3 = 1 and p_4 = 1000. The QP in Equation (14) was solved numerically using Matlab's quadprog function. In Figure 5 (scenario 1), the trajectory executed by the robot is shown in red and the obstacle in blue. In Figure 6 (scenario 1), the robot coordinates (x_g, y_g, ϕ_g) are shown in function of time. The results demonstrate that the robot reached the final position with good dynamic performance while avoiding the considered obstacle. Therefore, both the stabilization/tracking objectives and the safety constraints were satisfied. Figures 7 and 8 (scenario 2) show the results for the scenario in which the obstacle is far from the robot's trajectory. It can be observed that the robot follows the trajectory without any influence from the obstacle.

Figure 5. Scenario 1 – Trajectory executed by the robot in red and obstacle in blue.



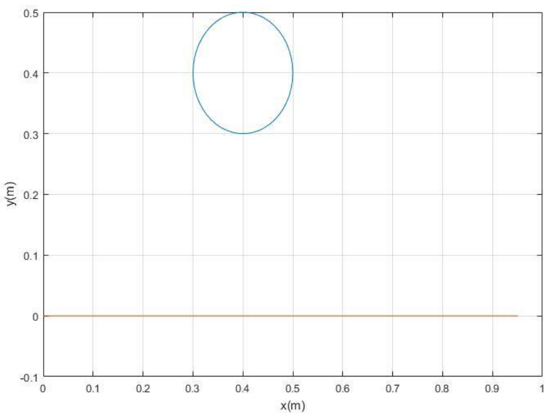
Source: Authors.

Figure 6. Scenario 1 – Robot coordinates (x_g, y_g, ϕ_g) in function of time.



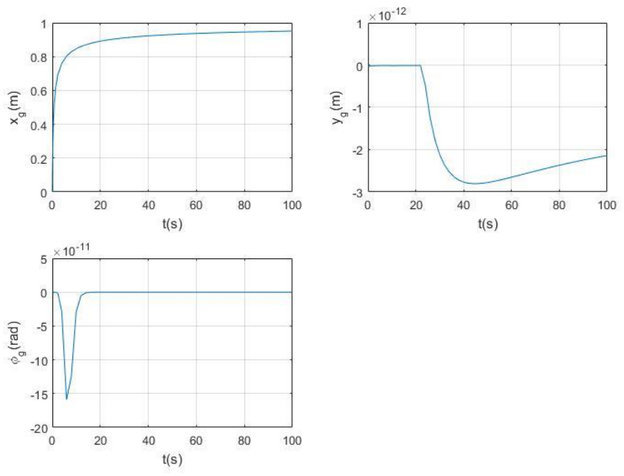
Source: Authors.

Figure 7. Scenario 2 – Trajectory executed by the robot in red and obstacle in blue.



Source: Authors.

Figure 8. Scenario 2 – Robot coordinates (x_g, y_g, ϕ_g) in function of time.

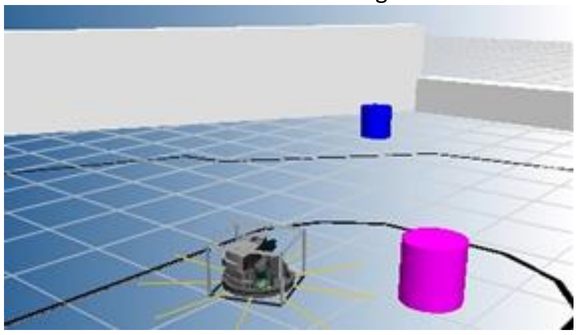


Source: Authors.

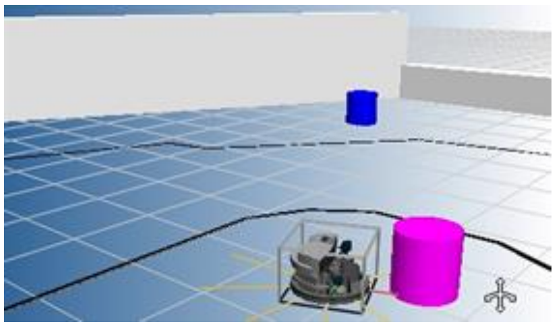
Posteriorly, results were obtained using the RobotinoSim simulator. As in the previous case, a simulation scenario was considered in which the robot must follow a reference trajectory starting from the initial position (x_i, y_i, ϕ_i) = (0 m, 0 m, 0 rad) and arriving at the final position (x_f, y_f, ϕ_f) = (1 m, 0 m, 0 rad). An obstacle is placed between the initial and final positions, and the robot must avoid it. Unlike the previous case, the obstacle position (x_o, y_o) is not known a priori; therefore, the robot's distance sensors were used to estimate the obstacle's position relative to the robot. The robot's position is estimated using the odometry block.

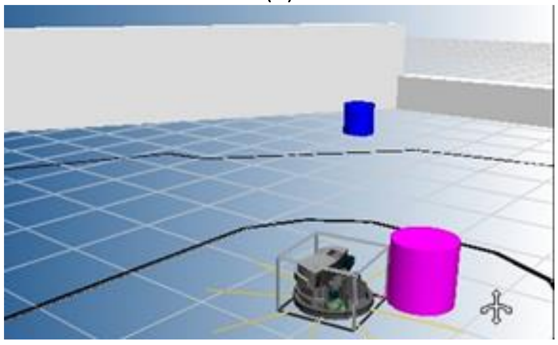
The results are shown in Figure 9. The simulation scenario presented corresponds to the standard environment generated by RobotinoSim. The figure shows two cylindrical obstacles, some black marks on the floor, and the mobile robot. It can be observed that the robot starts from an initial position with the odometer reset and avoids the purple obstacle. The yellow lines around the robot indicate the range of the distance sensors. When the sensor detects the obstacle, the sensor line turns red. As in the computational simulations with Matlab, the stabilization/tracking objectives and the safety constraints were satisfied.

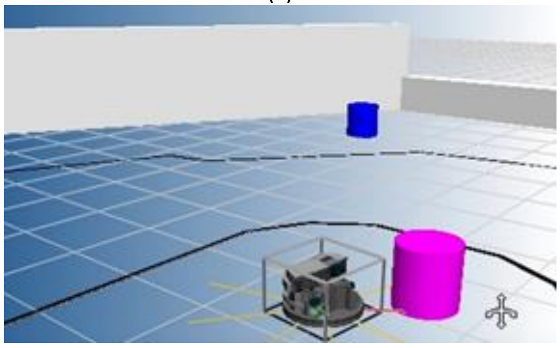
Figure 9. Simulation with RobotinoSim showing obstacle avoidance.

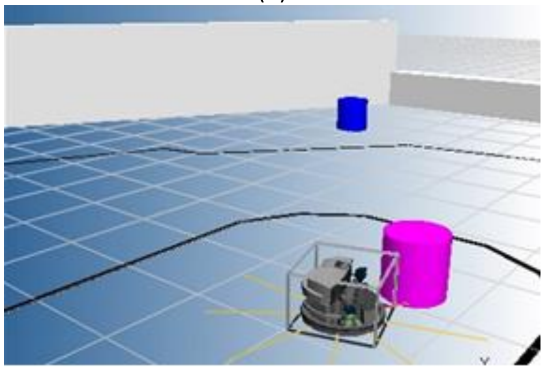


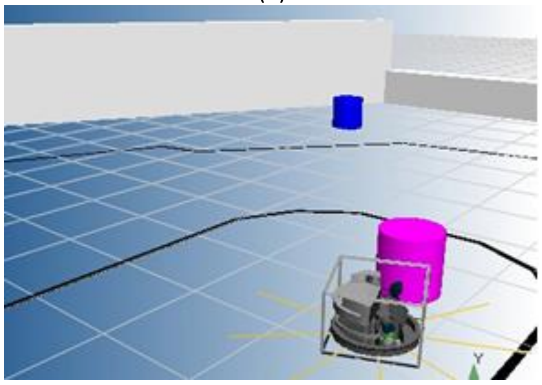
(a)


(b)


(c)


(d)

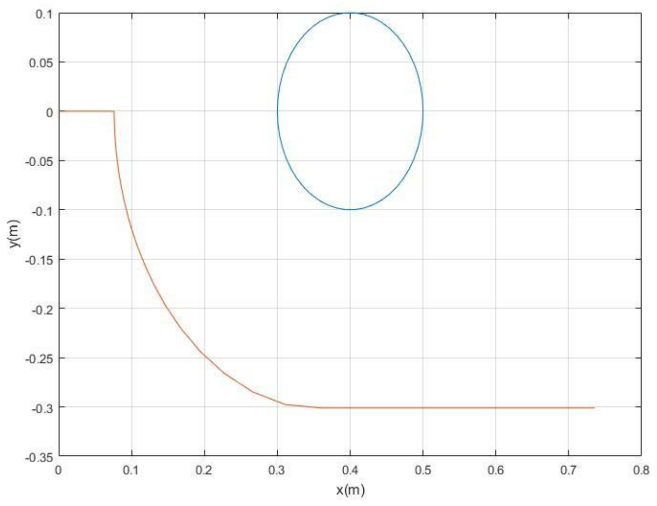

(e)


(f)

Source: Authors.

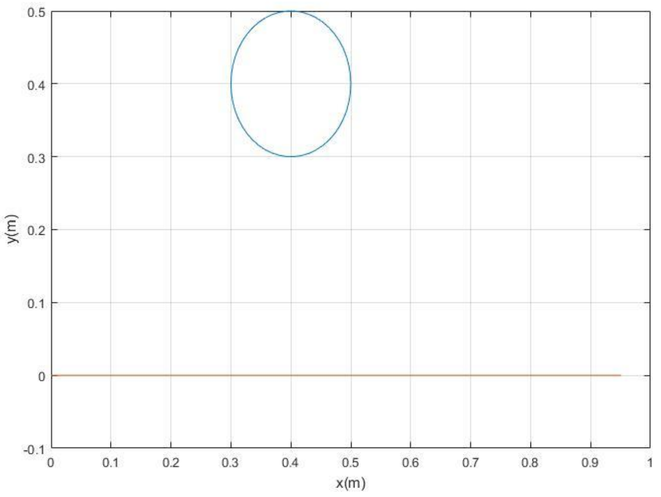
Some results were also obtained using the control framework in the form of Equation (6), i.e., when the stabilization/tracking objectives are described by a nominal control input u_{no} rather than a CLF *V(x)*. A constant u_{no} was considered, i.e., the mobile robot navigates at a constant velocity. The results for the same simulation scenario shown previously are presented in Figures 10 and 11. In Figure 10 (scenario 1), the trajectory executed by the robot is shown in red and the obstacle in blue. The results demonstrate that the robot reached the final position with good dynamic performance while avoiding the considered obstacle. Therefore, both the stabilization/tracking objectives and the safety constraints were satisfied. Figure 11 (scenario 2) shows the results for the scenario in which the obstacle is far from the robot's trajectory. It can be observed that the robot follows the trajectory without any influence from the obstacle. The idea is that, in the future, a linear or nonlinear nominal control law (such as PID or feedback linearization, for example) could be considered to further improve the dynamic performance of the stabilization/tracking objectives.

Figure 10. Scenario 1 (considering a nominal control input u_{no}) – Trajectory executed by the robot in red and obstacle in blue.



Source: Authors.

Figure 11. Scenario 2 (considering a nominal control input u_{no}) – Trajectory executed by the robot in red and obstacle in blue.



Source: Authors.

Conclusion

This work presents the control of the Robotino mobile robot, manufactured by Festo, which is used in several educational institutions in Brazil and worldwide, and unlike many didactic mobile robots, it features a more robust structure, higher actuation power, omnidirectional motion, a dedicated simulator, and can be programmed in multiple ways, making it more compatible with real-world applications. Several studies in the literature propose controlling Robotino to satisfy stabilization/tracking objectives; however, none of these studies apply CBFs to ensure that safety constraints are respected. Therefore, the main objective and contribution of this work is the application of a control framework that integrates stabilization/tracking objectives, described by a CLF or a nominal control law, and the safety constraints, described by CBFs, through a QP. The control framework ensures that the safety constraints take priority over the stabilization/tracking objectives.

The results were obtained through computational simulations using Matlab and RobotinoSim. The stabilization/tracking objectives, represented by reference trajectories, were described using a CLF as well as a nominal control law. The safety constraint, represented by position and coordinate constraints of the robotic system aimed at obstacle avoidance, was described by a CBF. The results demonstrated that both the stabilization/tracking objectives and the safety constraints were satisfied.

Suggestions for future work include the application of linear or nonlinear nominal control inputs (such as PID or feedback linearization, for example) to further improve the dynamic performance of the stabilization/tracking objectives, as well as the application of robust CBFs, i.e., CBFs capable of handling dynamic systems with model uncertainties, which is a topic that has been widely studied recently. More complex simulation scenarios with multiple obstacles can also be considered, for example. Furthermore, the proposed control framework can be extended to other robotic systems, such as robotic manipulators and quadcopters. Finally, it is important to highlight that, as mentioned earlier, several educational institutions in Brazil and worldwide have Robotino units; therefore, collaborations with these educational institutions could be established to obtain experimental results.

References

Ahmadi, M., Xiong, X., & Ames, A. (2022). Risk-averse control via CVaR barrier functions: application to bipedal robot locomotion. *IEEE Control Systems Letters*, *6*, 878-883. https://doi.org/10.1109/LCSYS.2021.3086854

Ames, A., Grizzle, J., & Tabuada, P. (2014). Control barrier function based quadratic programs with application to adaptive cruise control. *Proceedings of the IEEE Conference on Decision and Control (CDC)*, California, USA, 6271-6278. https://doi.org/10.1109/CDC.2014.7040372

Ames, A., Xu, X., Grizzle, J., & Tabuada, P. (2017). Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, *62*(8), 3861-3876. https://doi.org/10.1109/TAC.2016.2638961

Ames, A., Coogan, S., Egerstedt, M., Notomista, G., Sreenath, K., & Tabuada, P. (2019). Control barrier functions: theory and applications. *Proceedings of the European Control Conference (ECC)*, Naples, Italy, 3420-3431. https://doi.org/10.23919/ECC.2019.8796030

Borrelli, F., Bemporad, A., & Morari, M. (2017). *Predictive control for linear and hybrid systems*. Cambridge University Press. https://doi.org/10.1017/9781139061759

Ducaju, J., Olofsson, B., Robertsson, A., & Johansson, R. (2022). Robot cartesian compliance variation for safe kinesthetic teaching using safety control barrier functions. *Proceedings of the 18th International Conference on Automation Science and Engineering (CASE)*, Mexico City, Mexico, 2259-2266. https://doi.org/10.1109/CASE49997.2022.9926525

Elsayed, M., Hammad, A., Hafez, A., & Mansour, H. (2017). Real time trajectory tracking controller based on Lyapunov function for mobile robot. *International Journal of Computer Applications*, *168*(11), 1-6. https://doi.org/10.5120/ijca2017914540

El-sayyah, M., Saad, M., & Saad, M. (2025). Nonlinear model predictive control for trajectory tracking of omnidirectional robot using resilient propagation. *IEEE Access*, *13*, 112642-112653. https://doi.org/10.1109/ACCESS.2025.3583596

Ferraguti, F., Bertuletti, M., Landi, C., Bonfè, M., Fantuzzi, C., & Secchi, C. (2020). A control barrier function approach for maximizing performance while fulfilling to ISO/TS 15066 regulations. *IEEE Robotics and Automation Letters*, *5*(4), 5921-5928. https://doi.org/10.1109/LRA.2020.3010494

Gurriet, T., Singletary, A., Reher, J., Ciarletta, L., Feron, E., & Ames, A. (2018). Towards a framework for realizable safety critical control through active set invariance. *Proceedings of the ACM/IEEE International Conference on Cyper-Physical Systems (ICCPS)*, Porto, Portugal, 98-106. https://doi.org/10.1109/ICCPS.2018.00018

Hedman, M., & Mercorelli, P. (2021). FFTSMC with optimal reference trajectory generated by MPC in robust Robotino motion planning with saturating inputs. *Proceedings of the American Control Conference (ACC)*, New Orleans, USA, 1470-1477. https://doi.org/10.23919/ACC50511.2021.9482876

Igarashi, M., & Nakamura, H. (2018). Collision avoidance assist control for two-wheel vehicle robots by control barrier function. *Proceedings of the International Automatic Control Conference (CACS)*, Taoyuan, Taiwan. https://doi.org/10.1109/CACS.2018.8606761

Manjunath, A., & Nguyen, Q. (2021). Safe and robust motion planning for dynamic robotics via control barrier functions. *Proceedings of the IEEE Conference on Decision and Control (CDC)*, Austin, Texas, 2122-2128. https://doi.org/10.1109/CDC45484.2021.9682803

Mehra, A., Ma, W., Berg, F., Tabuada, P., Grizzle, J., & Ames, A. (2015). Adaptive cruise control: experimental validation of advanced controllers on scale-model cars. *Proceedings of the American Control Conference (ACC)*, Chicago, USA, 1411-1418. https://doi.org/10.1109/ACC.2015.7170931

Peng, C., Donca, O., Castillo, G., & Hereid, A. (2023). Safe bipedal path planning via control barrier functions for polynomial shape obstacles estimated using logistic regression. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, London, UK, 3649-3655. https://doi.org/10.1109/ICRA48891.2023.10160671

Pololu (2025, August 09). *Pololu 3pi+ 2040 Robot*. https://www.pololu.com/category/300/3pi-plus-2040-robot

Rauscher, M., Kimmel, M., & Hirche, S. (2016). Constrained robot control using control barrier functions. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, Korea, 279-285. https://doi.org/10.1109/IROS.2016.7759067

Robotino (2025, August 09). *Festo Robotino*. https://ip.festo-didactic.com/Infoportal/robotino/Overview/EN/index.html

Sa, M., Kotaru, P., & Sreenath, K. (2024). Point cloud-based control barrier function regression for safe and efficient vision-based control. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Yokohama, Japan, 366-372. https://doi.org/10.1109/ICRA57147.2024.10610647

Siegwart, R., Nourbakhsh, I., & Scaramuzza, D. (2011). *Introduction to autonomous mobile robots* (2nd ed.). MIT Press (MA). https://mitpress.mit.edu/9780262015356/introduction-to-autonomous-mobile-robots/

Singletary, A., Swann, A., Chen, Y., & Ames, A. (2022). Onboard safety guarantees for racing drones: high-speed geofencing with control barrier functions. *IEEE Robotics and Automation Letters*, *7*(2), 2897-2904. https://doi.org/10.1109/LRA.2022.3144777

Tang, Q., & Eberhard, P. (2013). Cooperative search by combining simulated and real robots in a swarm under the view of multibody system dynamics. *Advances in Mechanical Engineering*, *5*, 1-11. https://doi.org/10.1155/2013/284782

Toulkani, N., Abdi, H., Koskelainen, O., & Ghabcheloo, R. (2022). Reactive safe path following for differential drive mobile robots using control barrier functions. *Proceedings of the 10th International Conference on Control, Mechatronics and Automation (ICCMA)*, Belval, Luxembourg, 60-65. https://doi.org/10.1109/ICCMA56665.2022.10011466

Turtlebot3 (2025, August 09). *Robotis Turtlebot 3*. https://www.robotis.us/turtlebot-3/

Xu, X., Waters, T., Pickem, D., Glotfelter, P., Egerstedt, M., Tabuada, P., Grizzle, J., & Ames, A. (2017). Realizing simultaneous lane keeping and adaptive speed regulation on accessible mobile robot testbeds. *Proceedings of the IEEE Conference on Control Technology and Applications (CCTA)*, Hawai'i, USA, 1769-1775. https://doi.org/10.1109/CCTA.2017.8062713

Yang, T., Miao, Z., Yi, G., & Wang, Y. (2022). Safety-critical control of quadrotor UAVs with control barrier functions. *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Xishuangbanna, China, 1074-1079. https://doi.org/10.1109/ROBIO55434.2022.10011832