A Vision-Based Control Scheme for Safe Navigation in a Crowd

Paola Carboni[1], Giulia Nardini[1], Elisa Santini[1], Giovanbattista Gravina[1], Tommaso Belvedere[2], Michele Cipriano[1], Francesco D'Orazio[1], and Giuseppe Oriolo[1]

[1] Department of Computer, Control and Management Engineering, Sapienza University of Rome, Rome, Italy, {cipriano,dorazio,oriolo}@diag.uniroma1.it

[2] CNRS, Université de Rennes, Inria, IRISA, Campus de Beaulieu, 35042 Rennes Cedex, France, tommaso.belvedere@irisa.fr

Abstract. This paper proposes a vision-based control scheme for safe robot navigation in crowded environments. Unlike traditional methods relying on LiDAR or laser rangefinders, our approach leverages an RGB-D camera to capture rich visual information about the surroundings, allowing for a more comprehensive understanding of the scene. We address the challenge of predicting human motion in dynamic environments by combining a vision-based human detection module with a crowd prediction module. This allows the robot to anticipate potential collisions and generate safe motions. Additionally, we introduce an adaptive camera control strategy to enhance human detection performance by following their movement within the field of view of the camera. The proposed control scheme utilizes Control Barrier Functions (CBFs) to enforce safety constraints. By incorporating information about both robot-human relative position and velocity, CBFs ensure collision avoidance even in dynamic scenarios. The effectiveness of the method is evaluated by comparing the performance of different human detection algorithms, and by demonstrating the benefits of the adaptive camera control strategy and the overall safety achieved through the proposed vision-based control scheme.

1 Introduction

Nowadays, the growing demand for automation in service applications — such as healthcare, logistics and hospitality — has placed mobile robots at the forefront of research effort, due to their practically unlimited workspace and their ability to navigate in dynamic and densely populated environments. In these scenarios, ensuring safety of both the robot and the surrounding agents (e.g., humans) is the major challenge to face.

The fulfillment of this requirement primarily depends on the ability of the robot to gather real-time information about its surroundings, for instance, using onboard exteroceptive sensors. A 2D laser rangefinder is employed in [1] to acquire information about the surrounding positions of the agents, while [2] relies on LiDAR to perform obstacle detection. Despite offering precise distance

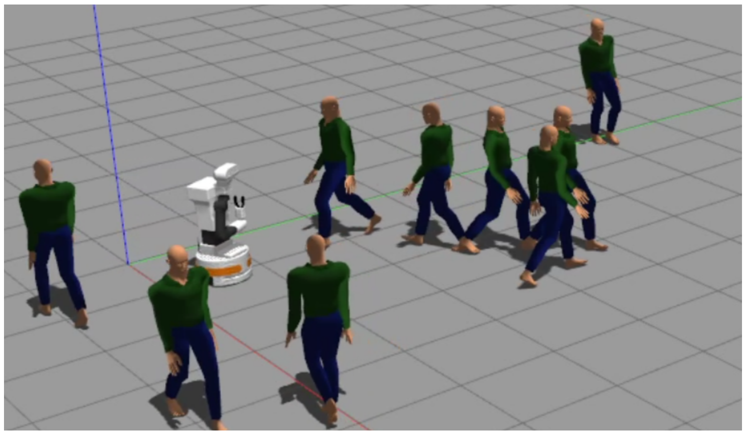

Fig. 1. Snapshot of the robot moving among humans in our simulation environment.

measurements, they lack visual context and require clustering algorithms to reconstruct actual objects from the generated point cloud. To tackle these issues, vision-based frameworks have emerged as a promising solution for enabling safe navigation. This development is also motivated by the availability of powerful object detection systems that allow real-time image processing, providing rich and detailed information about the surrounding environment. Among vision-based sensors, RGB-D cameras enable accurate object recognition and 3D reconstruction leveraging color and depth information [3], [4].

In addition to the adopted perception methodologies, a key aspect in enhancing safety in dynamic scenarios is the ability to predict the future motion of agents in the environment to anticipate potential collisions. Indeed, collision avoidance constraints formalized relying solely on distance-based information, as seen in [5] and [6], may not be sufficient for ensuring safety in dynamic environments. To tackle this issue, [7] proposes a Reinforcement Learning approach to learn socially cooperative policies. However, while significantly improving the ability of the robot to anticipate human movements, such data-driven models often require a training phase and suffer from limited generalization power. Recently, Control Barrier Functions (CBFs) [8] have been successfully exploited to enforce safety, as they allow for the formulation of collision avoidance constraints that also account for the robot-obstacle relative velocity, as demonstrated in [9].

In this work we present a vision-based pipeline for safely controlling a robot moving in a crowd. We test how the performance of the method is affected by the detection algorithm by comparing two state-of-the-art implementations, and we devise an adaptive camera control strategy to enhance the detection performance by following the movement of the humans, demonstrating its effectiveness. Finally, the motion of the humans at risk of collision is estimated and fed to a QP controller achieving obstacle avoidance through CBF constraints.

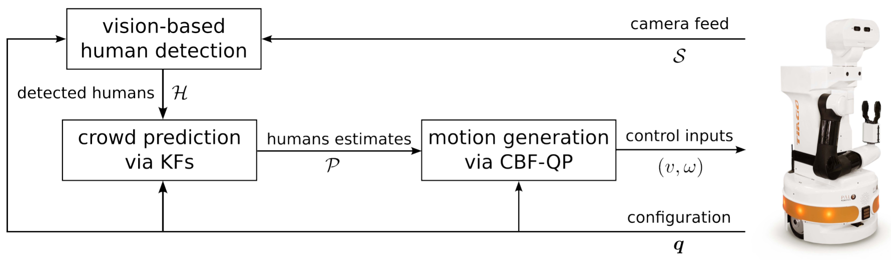

Fig. 2. Block diagram of the proposed framework.

2 Problem Formulation

A wheeled mobile robot equipped with an RGB-D camera is assigned the task of navigating an environment populated by a crowd of humans. In order to achieve this goal while ensuring safety, the robot must reconstruct the positions of the humans and generate a motion free of collisions.

More specifically, safe navigation of a robot moving in a crowd can be formulated as the problem of generating commands such that:

i. the resulting motion is feasible with respect to the robot kinematics and the input constraints (such as wheel velocity limits);
ii. the robot avoids collisions with humans at all times while it is moving;
iii. the robot executes the desired task (set-point regulation or trajectory tracking) with minimal violation.

To meet the aforementioned requirements, the system needs to be able to detect the humans in its surroundings and accurately estimate their position and velocity in real time. It is reasonable to assume that only a subset of the humans moving in the environment poses a challenge to the robot's safe navigation. Therefore, a priority mechanism needs to be introduced to select which humans estimate should be computed at any given time. Additionally, since human movement is estimated using the vision sensor, it is important to address the limitations on the Field of View (FOV) of the camera and take steps to improve the reliability of the detection. This can be achieved, for example, by using the pan-tilt action of the camera to keep the closest humans in the FOV of the robot.

3 Proposed Method

We introduce a vision-based framework for safe robot navigation in a crowd. Our proposed method is able to generate motions to solve the problem described in Sect. 2, and is composed of three components: a vision-based human detection module, a crowd prediction module and a motion generation module, as depicted in Fig. 2.

We discuss the application to a differential-drive robot, which is kinematically equivalent to a unicycle, although the method can be easily extended to other mobile robots. Let $\boldsymbol{p}_r = (x, y)$ be the Cartesian position and θ its orientation, so that $\boldsymbol{q} = (x, y, \theta) \in SE(2)$ describes the configuration of the robot. Then, having as inputs $\boldsymbol{u} = (v, \omega)$ the driving and steering velocity respectively, the kinematic model of the robot is

$$\dot{\boldsymbol{q}} = \begin{pmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{pmatrix} .$$

The robot commands are computed by the motion generation module. Here, the action of a nominal controller is filtered by a safety layer which uses the latest estimate of the human position and velocity to guarantee that the resulting motion is collision-free.

The human detection and crowd prediction modules are responsible for estimating the position and velocity of each human in the FOV from the image provided by the RGB-D camera. This process can be divided into two steps: (i) detect all humans within the FOV of the camera and reconstruct their position, (ii) use a human prediction module capable of generating position and velocity estimates for all the humans at risk of collision.

3.1 Vision-based human detection

In our framework, human detection is performed by either using classical computer vision algorithms based on Histogram of Gradients (HoG) features [10], or by using YOLO-v8 [11], a deep neural network architecture for real-time object detection. The first algorithm works by extracting features of humans using AdaBoost, identifying the appropriate blocks of the image containing the humans from a predefined set of possible blocks. The second algorithm is a neural network trained on COCO [12] for detection, and ImageNet [13] for classification.

The detection algorithm is provided with the camera feed $\mathcal{S}$, and it returns a list of bounding boxes, each one containing a detected human. The center of each bounding box $\boldsymbol{h}_i^{\text{img}}$ is then used to determine the Cartesian position $\boldsymbol{h}_i \in \mathbb{R}^2$ of each human, which can be derived through the pinhole camera model [14] using the following:

$$\boldsymbol{h}_i = \Pi_{xy}\left(\boldsymbol{R}_c d(\boldsymbol{h}_i^{\text{img}})\boldsymbol{K}^{-1}\boldsymbol{h}_i^{\text{img}} + \boldsymbol{p}_c\right) ,$$

where $\boldsymbol{R}_c$ and $\boldsymbol{p}_c$ are, respectively, the orientation and position of the camera in the world frame, Π_{xy} is a function extracting the x and y components of the argument, $\boldsymbol{K}$ is the camera matrix, defined as

$$\boldsymbol{K} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} ,$$

with (f_x, f_y) focal lengths and (c_x, c_y) principal point offset, and $d(\cdot)$ is a function mapping a point in the image plane to a distance (whose value is retrieved in practice using the depth sensor).

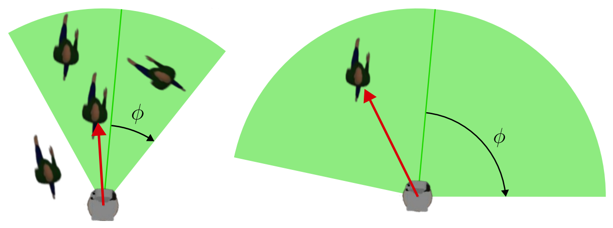

Fig. 3. Top-view representation of the range adaptation strategy. Red arrows indicate the humans selected as the most dangerous.

In the following, we will denote as $\mathcal{H}$ the set of all detected humans, which is defined as

$$\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N\},$$

with N total number of detections.

Camera position control To further improve the detection performance, the camera, which is assumed to be mounted on a pan-tilt platform, can be moved to point toward humans that are identified as dangerous. We define a conic region on the ground attached to the base frame of the robot and select as most dangerous the closest human in the region (cfr. Fig. 3). The gaze of the camera will, hence, be oriented to keep the closest human in the center of the image plane. The size of the cone is determined by an angle ϕ.

Our camera control strategy is adaptive, as the angle ϕ is computed online, starting from a predefined value, to trade off between following the closest human and keeping the direction of motion within the FOV of the camera. When many humans are present in the scene, the cone is progressively tightened depending on the number of detected humans to isolate those that are more dangerous since they are in front of the robot. Otherwise, when just one human is detected, the range is widened to allow for more humans to be considered dangerous, which makes it possible to also take into account humans that are approaching from the sides and might therefore become dangerous.

3.2 Crowd prediction via KFs

Once the set of positions of the humans $\mathcal{H}$ is reconstructed from the camera, it is fed to a crowd prediction module based on Kalman Filters (KFs), which has the objective of estimating the current position and velocity of the humans closest to the robot. We closely follow the approach described in [1], in which the M closest measurement from $\mathcal{H}$ are processed. The parameter M determines the size of the pool of KFs considered in the module and can be tuned accordingly to the size of the crowd and of the computational resources. First, we perform a closest-distance data association between past estimates and current measurements, then we feed the measurements to the associated KFs which, using a double integrator prediction model, estimate the current position and velocity of the humans, denoted as $\hat{\boldsymbol{h}}_i$ and $\hat{\boldsymbol{v}}_i$ respectively, which are collected in a set

$$\mathcal{P} = \left\{ \hat{\boldsymbol{h}}_1, \hat{\boldsymbol{v}}_1 \ldots, \hat{\boldsymbol{h}}_M, \hat{\boldsymbol{v}}_M \right\},$$

and fed to the controller to perform obstacle avoidance.

Each KF regulates its operation through a Finite State Machine (FSM), which consists of four states: *Idle*, *Start*, *Active* and *Hold*. In the following, we briefly describe the states of each FSM. For a detailed explanation, we invite the reader to refer to [1].

- *Idle.* The KF is inactive. As soon as a new measurements is received, the FSM state becomes *Start*.
- *Start.* If a new measurement is available, the KF initializes the state estimate, and the FSM state becomes *Active*. The FSM state becomes *Idle* if there is no new measurement.
- *Active.* The state estimates are updated using new measurements as long as they are available. Whenever this is not true, the FSM state becomes *Hold*.
- *Hold.* The KF remains in this state for a limited time, updating the estimates with the last available measurement. If a new measurement is found, the FSM state becomes *Active*. If the time runs out, the FSM state becomes *Idle* and the estimate is discarded.

3.3 Safe motion generation via CBF-QP

The robot is commanded using velocity inputs obtained from the solution of a Quadratic Program (QP), which employs the well-established framework of Control Barrier Functions (CBF), designed to minimally modify the action of a nominal controller to guarantee safety [15]. Such nominal controller can be designed separately to achieve the desired task in the absence of humans, and can therefore employ any classic control law. Alternatively, it would be possible to include in the QP also the generation of a command to perform the main task by employing the CLF-CBF-QP framework [8].

Considering a simple reactive QP controller, as opposed to a predictive controller as in [1], makes the computational requirements of the proposed method particularly modest, freeing up resources and lowering the bar for implementation on real robots.

In our setting, we impose M CBF constraints designed to avoid collision with the currently estimated humans, whose position and velocity are reconstructed by the crowd prediction module, while guaranteeing a distance not lower than a predetermined safe distance at all times. Let $\boldsymbol{\xi}_i = (\boldsymbol{q}, \hat{\boldsymbol{h}}_i)$ be the combined configuration of the robot and the i-th human in $\mathcal{P}$. We determine that a state is *safe* if the distance between the human and the robot is larger than a safety distance $\rho > 0$ computed considering the dimension of the robot bounding circle plus a safety margin. Accordingly, we can define the safe set as

$$\mathcal{C}_i = \left\{ \boldsymbol{\xi}_i \in SE(2) \times \mathbb{R}^2 \, : \, g(\boldsymbol{\xi}_i) \geq 0 \right\},$$

where

$$g(\boldsymbol{\xi}_i) = \|\boldsymbol{p}_r - \hat{\boldsymbol{h}}_i\| - \rho,$$

which is a valid CBF. The set $\mathcal{C}_i$ can then be made forward invariant if the constraint

$$\dot{g}(\boldsymbol{\xi}_i, \hat{\boldsymbol{v}}_i, \boldsymbol{u}) \geq -\alpha g(\boldsymbol{\xi}_i) \tag{1}$$

is satisfied for some $\alpha > 0$. Note how the velocity $\hat{\boldsymbol{v}}_i$ of the human appears in the CBF constraint, which explains our need for its estimate.

When multiple humans are to be avoided, the robot has to remain in the intersection of the safe sets $\mathcal{C}_i$ for each robot-human pair. In this case, we guarantee safety by enforcing CBF constraints (1) for every pair.

Two different nominal controllers generating command $\boldsymbol{u}_{\mathrm{ff}}$ to fulfill the main task are considered: a posture regulation controller based on polar coordinates and a trajectory tracking controller based on input-output feedback linearization [16]. At each instant, $\boldsymbol{u}_{\mathrm{ff}}$ is modified to guarantee safety by solving the CBF-QP problem:

$$\begin{aligned} \min_{\boldsymbol{u} \in [\boldsymbol{u}_{\min}, \boldsymbol{u}_{\max}]} \quad & ||\boldsymbol{u} - \boldsymbol{u}_{\mathrm{ff}}||^2 \\ \text{subject to} \quad & \dot{g}(\boldsymbol{\xi}_i, \hat{\boldsymbol{v}}_i, \boldsymbol{u}) \geq -\alpha g(\boldsymbol{\xi}_i) \quad \text{for } i = 1, \ldots, M \end{aligned} \tag{2}$$

Consistently with the output of the crowd prediction module, if one of the KFs is in an idle state, the corresponding CBF constraint is deactivated. If no humans state are being estimated, the solution of QP (2) results in a minimal modification of the nominal action to satisfy input constraints. In case a human is moving towards the robot at high speed, such that the collision avoidance constraint cannot be satisfied given the velocity limitations of the robot, an emergency brake command is triggered to minimize the risk of an actual collision. Once the QP finds a new solution the motion is restarted.

4 Simulations

The performance of the method has been evaluated in simulation comparing three different scenarios using TIAGo robot. All simulations have been conducted in Gazebo on an Intel Core i9-9900K CPU at 3.60 GHz.

The robot base is commanded with driving and steering velocities, which are limited by the low-level controller on the robot. CBF-QP is always solved within 1 millisecond, making it possible to run the framework in real-time. The control architecture runs at 50 Hz, with the KFs always utilizing the latest available measurement computed by the human detection module, running at 30 Hz. The number of KFs is set to $M = 3$, the camera FOV mounted on the robot is $63°$, the predefined value of the angle defining the cone is $\phi = 60°$, the parameter of the CBF condition is $\alpha = 0.5$, the safety distance for the CBF is $\rho = 1.3$ meters. In the current implementation, the pan-tilt action is achieved by pointing the camera at the current location of the most dangerous human through a specific function implemented on our robotic system, however, it would also be possible

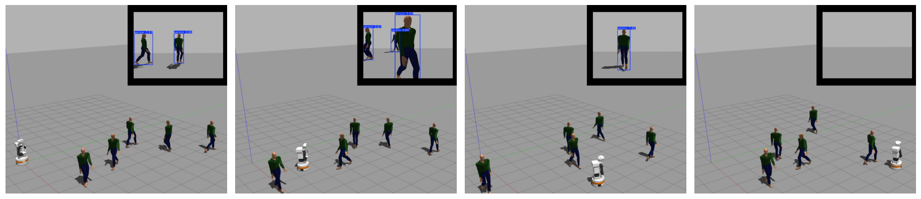

Fig. 4. Snapshots of TIAGo moving in a crowd of 5 humans, including the camera image showing the human detection at work.

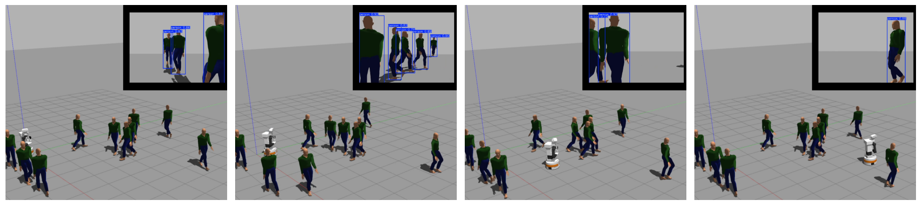

Fig. 5. Snapshots of TIAGo moving in a crowd of 10 humans, including the camera image showing the human detection at work.

to follow standard visual servoing techniques such as the one described in [17]. Note that the humans considered in the scene are not robot-friendly, in the sense that they are not aware of the presence of the robot. Moreover, their motion is assumed to be omni-directional.

Figures 4 and 5 show the robot safely moving in a crowd, detecting humans and avoiding them. We invite the reader to watch the accompanying video[3], which shows different simulations of the robot moving in a crowd using the posture regulation controller, as well as additional simulations where the trajectory tracking controller is used.

Table 1 evaluates the framework when using the posture regulation controller, comparing its behavior in two different environments (the first one populated by 5 humans, the second one populated by 10 humans) using different human detection algorithms, and optionally considering the adaptive camera control strategy described in the previous section. The success rate of simulations that are concluded without any collision occurring is averaged over 10 simulations for each scenario. The results clearly show that the HoG detector is not able to provide a sufficient detection accuracy. We have indeed observed that, in our conditions, the range in which a human is detected is rather limited and that detection often fails when humans are only partially observed or are viewed from the side. When the camera control is deactivated, it becomes essential to increase the *persistency* of the human estimate in the crowd prediction module to mitigate the effect of the frequent loss of detection. Still, only in the simplest 5 human scenario, there have been a few instances in which the robot is able

[3] `https://youtu.be/-NBDeRpRQ9w`

Detection	Camera control	Success rate	
		5 humans	10 humans
HoG	Yes	30%	0%
	No	0%	0%
YOLO-v8	Yes	90%	90%
	No	40%	30%

Table 1. Performance evaluation of the proposed framework over two different scenarios (5 and 10 humans), with two detection algorithms, with and without the inclusion of our camera control strategy.

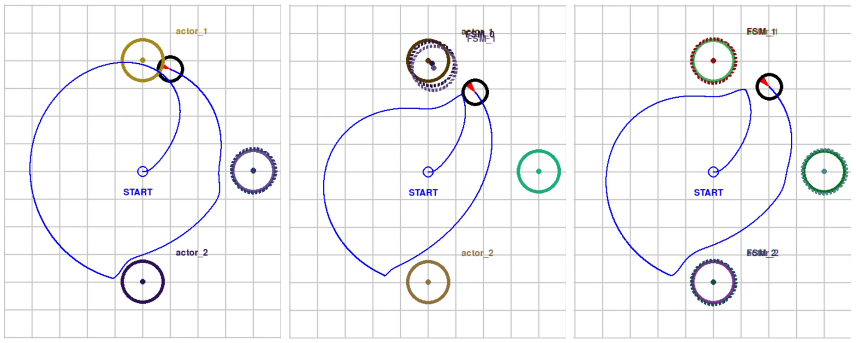

Fig. 6. Motion executed by the robot when tracking a circle, using HoG to detect humans on the left, YOLO-v8 on the center, and the ground truth on the right.

to safely reach the goal, with a 30% success rate. On the other hand, YOLO-v8 has proven to be much more successful at detecting humans over an extended distance range and also when only partially observed. Nevertheless, the success rate becomes acceptable only when the camera control strategy is introduced, likely due to the limited FOV of the on-board camera, which severely limits the detection ability when humans are close to the side of the robot.

Concerning the trajectory tracking task, we report in Fig. 6 the results obtained with a circular trajectory in an environment populated by 3 humans. In this case, the robot is not able to avoid the first human when using the HoG detector. On the other hand, it is possible to appreciate how the tracking of the desired trajectory is close to the ground truth case (i.e., when the positions of all humans are known a priori) when detecting humans using YOLO-v8, further demonstrating its high level of accuracy.

We have compared our method based on CBF-QP for collision avoidance with a QP which considers a purely distance-based constraint, finding out that the latter has a lower success rate. This result is consistent with the one obtained in [1], proving once again that the CBF are able to anticipate the activation of the constraint.

5 Conclusions

In this work, we demonstrated the performance of our vision-based pipeline for safely navigating a robot through a human crowd. Results show the importance of exploiting the pan-tilt action of the camera to maximize the detection reliability and thus safety. Future work will entail

– the experimental validation of the method,
– the further development of the perception module by fusing camera and rangefinder measurements,
– the introduction of fixed non-human obstacles in a more realistic scenario,
– the study of different human motion prediction strategies suitable for predictive controllers.

References

1. V. Vulcano, S. G. Tarantos, P. Ferrari, and G. Oriolo, "Safe robot navigation in a crowd combining NMPC and control barrier functions," in *2022 IEEE 61st Conf. on Decision and Control*, pp. 3321–3328, 2022.
2. F. Gao, C. Li, and B. Zhang, "A dynamic clustering algorithm for lidar obstacle detection of autonomous driving system," *IEEE Sensors Journal*, vol. 21, pp. 25922–25930, 2021.
3. M. Skoczeń, M. Ochman, K. Spyra, M. Nikodem, D. Krata, M. Panek, and A. Pawłowski, "Obstacle detection system for agricultural mobile robot application using RGB-D cameras," *Sensors*, vol. 21, no. 16, 2021.
4. Z. Xu, X. Zhan, Y. Xiu, C. Suzuki, and K. Shimada, "Onboard dynamic-object detection and tracking for autonomous robot navigation with RGB-D camera," *IEEE Robotics and Automation Letters*, vol. 9, pp. 651–658, 2024.
5. B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora, "Model predictive contouring control for collision avoidance in unstructured dynamic environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4459–4466, 2019.
6. H. Febbo, P. Jayakumar, J. L. Stein, and T. Ersal, "Real-Time Trajectory Planning for Automated Vehicle Safety and Performance in Dynamic Environments," *Journal of Autonomous Vehicles and Systems*, vol. 1, p. 041001, 12 2021.
7. C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6015–6022, 2019.
8. A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European Control Conference (ECC)*, pp. 3420–3431, 2019.
9. A. Thirugnanam, J. Zeng, and K. Sreenath, "Safety-critical control and planning for obstacle avoidance between polytopes with control barrier functions," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 286–292, 2022.
10. Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan, "Fast human detection using a cascade of histograms of oriented gradients," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, pp. 1491–1498, 2006.
11. G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLO," Jan. 2023.

12. T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: common objects in context,” in *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V* (D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), vol. 8693 of *Lecture Notes in Computer Science*, pp. 740–755, Springer, 2014.
13. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.
14. R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. 2 ed., 2003.
15. A. Singletary, K. Klingebiel, J. Bourne, A. Browning, P. Tokumaru, and A. Ames, “Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, p. 8129–8136, IEEE, Sept. 2021.
16. B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer, 2009.
17. L. Freda and G. Oriolo, “Vision-based interception of a moving target with a non-holonomic mobile robot,” *Robotics and Autonomous Systems*, vol. 55, p. 419–432, June 2007.