2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Safe Control using Vision-based Control Barrier Function (V-CBF)

Hossein Abdi, Golnaz Raja, and Reza Ghabcheloo

Abstract— **Safe motion control in unknown environments is one of the challenging tasks in robotics, such as autonomous navigation. Control Barrier Function (CBF), as a strong mathematical tool, has been widely used in many safety-critical systems to satisfy safety requirements. However, there are only a handful of recent studies on safety controllers with perception inputs. Common assumptions in most of the works are that the CBF is already known and obstacles have predefined shapes. In this work, we introduce a novel Vision-based Control Barrier Function (V-CBF), which enables generalization to new environments and obstacles of arbitrary shapes. We then derive CBF safety conditions over RGB-D space and relate those to actual robot control inputs. To train the CBF function, we introduce a method to generate ground truth with desired properties complying with CBF and a method to generate part of the CBF as an image-to-image translation problem. We finally demonstrate the efficacy of V-CBF on the safe control of an autonomous car in CARLA simulator.**

I. INTRODUCTION

Safe control and navigation of autonomous systems in complex and unknown environments is one of the most challenging issues in today's robotics [1]. Safe control refers to devising control inputs to guarantee that: i) states of the dynamic system never reach the unsafe region; ii) control inputs never violate the physical constraints of the system. In recent years, Control Lyapunov Function (CLF) and Control Barrier Function (CBF) have attracted considerable interest due to their strong mathematical guarantee in satisfying stability and safety requirements. Control barrier functions are used in different application areas such as autonomous cars [2], [3]; multi-robot systems [4], [5]; quadrotors [6], [7]; and legged Robots [8], [9].

Integration of perception systems into robot control in a rigorous manner is a hot topic. Authors in [10] and [11] integrate CBF to visual servoing to maintain visibility to the target. The visibility constraints are defined as a control barrier function in the image frame as a circle. Similarly, [12] use CBF to maintain visibility to a target, in addition to maintaining communication and avoiding collision in a multi-UAV setting.

A common assumption in all the previous works is that the control barrier function is already known while navigation of mobile robots in unknown ambient requires online detection of unsafe regions using onboard sensing, and therefore, the CBF should be constructed online. For a thorough survey, please refer to [13].

This work was supported by Academy of Finland (SA) 345517, under SA-NSF joint call on artificial intelligence and wireless communication.

All authors are with the Faculty of Engineering and Natural Sciences, Tampere University, Finland. {hossein.abdi, golnaz.raja, reza.ghabcheloo}@tuni.fi

A theoretical technique addressing the CBF synthesis is introduced in [14], where the authors create a permissive barrier certificate using the sum-of-squares method. A solution is provided in [15], where a so-called BarrierNet is introduced, a network trained that takes image as input and outputs parameters of a class of CBFs, in their case circular obstacles. The authors in [2] use BarrierNet to design safe controllers in lane-keeping and obstacle avoidance for autonomous vehicles. In [16], CBFs are used to train safe controllers (policies) in an imitation learning setting.

In [17], the authors utilize feedback from a 2D LiDAR sensor and train neural networks to learn an observation-based control barrier function and maintain safety during navigation. In [18], the authors constructed the CBFs again using a LiDAR scanner. The core of their method is in learning an approximate of sign-distance-function to obstacles given Lidar input, and inclusion of the function error in CBF formulation. Another related study pertaining to the synthesis of the control barrier function is presented in [19], where the authors utilize a support vector machine classifier to generate the desired CBF based on the sets of safe and unsafe states derived from LiDAR measurements.

In this paper, we introduce a Vision-based Control Barrier Function (V-CBF), to address the safe control of a robot. Our contributions are as follows: 1) a novel vision-based approach to constructing CBF as a function of observations (RGB-D image); 2) a CBF enabling generalization to new environments and obstacles of arbitrary shapes; 3) derivation of CBF conditions over image space for robot kinematics model (e.g. relative degree one) with linear and angular speeds as control inputs; 4) a method to generate ground truth with desired properties complying with CBF; 5) a method to generate part of the CBF as an image-to-image translation problem; 6) demonstration of the efficacy of the method on safe control of an autonomous car in CARLA simulator.

The paper is organized as follows. Section II summarizes the principal concept of the control barrier function and some mathematical definitions. Then, in section III, we develop our control methodology by introducing the V-CBF approach. We then discuss the technique of generating the CBF by RGB-D image using a conditional Generative Adversarial Network (cGAN) in section IV. Section V is dedicated to demonstrating the proposed control strategy's performance by implementing it on the CARLA simulator. And finally, in Section VI, we provide our conclusions.

979-8-3503-2365-8/23/$31.00 ©2023 IEEE

II. Preliminaries and Background

Considering a closed-loop non-linear affine control system:

$$\begin{aligned} \dot{\boldsymbol{x}} &= f(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u} \\ \zeta &= q(\boldsymbol{x}, \boldsymbol{m}) \end{aligned} \tag{1}$$

where $\boldsymbol{x} \in \mathbb{X} \subseteq \mathbb{R}^n$ is state vector, $\boldsymbol{u} \in \mathbb{U} \subseteq \mathbb{R}^m$ is control input. Also, $\zeta \in \mathbb{R}^{W \times H \times C}$ is the observation which is the captured RGB-D image where W, H, C denote width, height, and number of channels. Also, the map q generates this image as a function of states of the system and environment variables $\boldsymbol{m}$. Furthermore, $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ are locally Lipschitz continuous functions.

Definition 1: A function $f : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^m$ is locally Lipschitz continuous at an arbitrary point $x \in D$ if there exist constants $M > 0$ and $\delta > 0$ such that $\|f(x) - f(x')\| \leq M\|x - x'\|$ holds for all $\|x - x'\| \leq \delta$.

In this study, safe control refers to devising control inputs to guarantee that: i) states of the dynamic system never reach the unsafe region; ii) control inputs never violate the physical constraints of the system.

Definition 2: The system (1) is considered safe if $\boldsymbol{x}(t) \in \mathbb{X}_s \subseteq \mathbb{X}$, $\boldsymbol{u}(t) \in \mathbb{U}_s \subseteq \mathbb{U}$, $\forall t \geq 0$. Where $\mathbb{X}_s$ and $\mathbb{U}_s$ are set of safe states and safe control inputs, respectively.

Let $S \subseteq \mathbb{X}_s$ be zero-superlevel set of a smooth and continuously differentiable function $h(\boldsymbol{x}) : \mathbb{X} \rightarrow \mathbb{R}$ that satisfies the following conditions:

$$S = \{\boldsymbol{x} \in \mathbb{X} \mid h(\boldsymbol{x}) \geq 0\} \tag{2a}$$

$$\partial S = \{\boldsymbol{x} \in \mathbb{X} \mid h(\boldsymbol{x}) = 0\} \tag{2b}$$

$$Int(S) = \{\boldsymbol{x} \in \mathbb{X} \mid h(\boldsymbol{x}) > 0\} \tag{2c}$$

where ∂S is the set boundary, and $Int(S)$ is interior of set S. To guarantee that system (1) will remain safe for all time, we should make the set S a forward invariant set.

Definition 3: An arbitrary set $A \subseteq \mathbb{R}^n$ is called forward invariant set for the dynamic system $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$ if for any $\boldsymbol{x}(0) \in A$, it satisfies $\boldsymbol{x}(t) \in A$, $\forall t \geq 0$ [20].

It is mathematically proven [21] that the function $h(\boldsymbol{x})$ is a control barrier function if there exists an extended class κ function $\alpha(.)$ such that for the control system (1):

$$\sup_{\boldsymbol{u} \in \mathbb{U}} [L_f h(\boldsymbol{x}) + L_g h(\boldsymbol{x})\boldsymbol{u}] \geq -\alpha(h(\boldsymbol{x})) \tag{3}$$

where $\dot{h}(\boldsymbol{x}, \boldsymbol{u}) = L_f h(\boldsymbol{x}) + L_g h(\boldsymbol{x})\boldsymbol{u}$ and L_f, L_g denote Lie derivatives. Therefore, satisfying this condition will make safe set S forward invariant; thus, for all $\boldsymbol{x}(0) \in S$, trajectories will remain inside S.

Definition 4: A continuous function $\alpha(.) : [0, a) \rightarrow [0, \infty)$, $a > 0$ belongs to class κ function if it is strictly increasing and $\alpha(0) = 0$. Moreover, a continuous function $\beta(.) : \mathbb{R} \rightarrow \mathbb{R}$ is said to belong to extended class κ function if it is strictly increasing and $\beta(0) = 0$.

Definition 5: The common notation $L_\xi \eta(x)$ is utilized for the Lie derivative of $\eta(x)$ along the vector field $\xi(x)$ i.e. $L_\xi \eta(x) = \frac{\partial \eta(x)}{\partial x} \xi(x)$.

III. Control Methodology

A. Vision-based Control Barrier Function (V-CBF)

As is shown in Fig. 1, consider the right-handed 3D Cartesian frames $\mathcal{F}_w$, $\mathcal{F}_b$, $\mathcal{F}_c$, $\mathcal{F}_i$, and $\mathcal{F}_p$ as world, body, camera, image, and pixel coordinate systems, respectively. We define the safety using control barrier function as follows:

$$h(\zeta) = h_{img}(u, v) + h_d(d) \tag{4}$$

where $h_{img}(u, v) : \mathbb{R}^2 \rightarrow \mathbb{R}$ is a scalar function defined on pixel coordinate, and $h_d(d) : \mathbb{R} \rightarrow \mathbb{R}$ is a function defined on depth domain, which d denotes the depth value at the pixel (u, v).

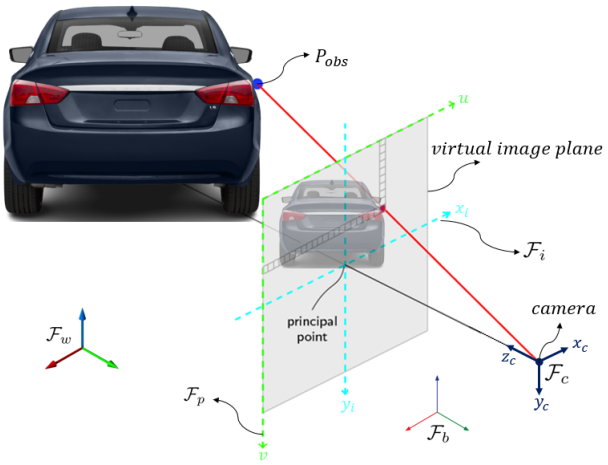

Fig. 1. Coordinate systems definition

We assume that we can segment the RGB-D image ζ to safe and unsafe regions. We then learn the contour of $h_{img}(.)$ using the known segmentation ground truth, and design $h_d(.)$ as a strictly increasing function, such that the final constructed $h(\zeta)$ has the required properties of CBF defined in (2).

To train $h_{img}(.)$, we will create a ground truth such that $h_{img}(u, v)$ is zero on the borders of the unsafe regions, and continuously increase from negative values inside the unsafe regions to positive values outside them. The later property prevents gradient of h from vanishing $\partial h_{img}/\partial u = 0$, $\partial h_{img}/\partial v = 0$.

In addition, as far as the robot and obstacle have a safe distance, the final constructed CBF $h(\zeta)$ should hold $h(\zeta) > 0$. In this regard, to avoid $h(\zeta) < 0$, when $h_{img}(u, v) < 0$, the value of $h_d(.)$ should be more than the magnitude of $h_{img}(.)$. Obviously, it is because of this fact that $h_d(.)$ is ever

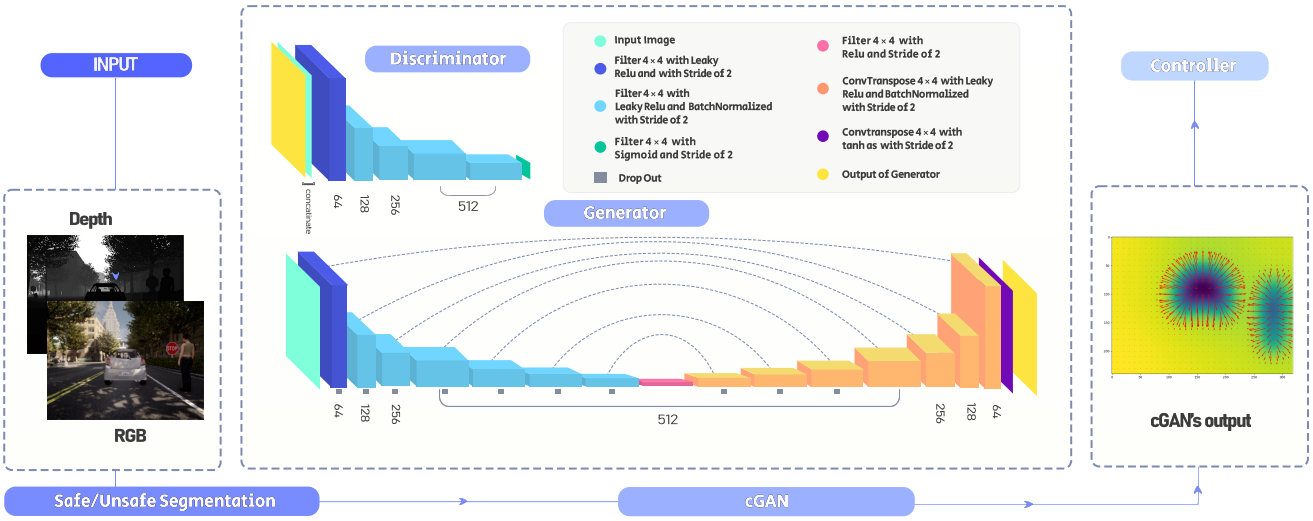

Fig. 2. Framework of safe control using V-CBF. The RGB-D image feeds to the network as input. In the first stage, an instance segmentation algorithm classifies the pixels into safe and unsafe categories. The preprocessed image goes through a cGAN to produce the contour and gradient of h_{img}. At the end of the procedure, the control input is generated by V-CBF.

positive while $h_{img}(.)$ has negative values in some zones. To this aim, we restrict the values of h_{img} between an upper and lower bound; and choose the $h_d(.)$ as a strictly increasing function so that $h_d(0) = 0$. Simply, it can be demonstrated that the value of $h(\zeta)$ will be zero in a safe distance (d_s) to the border of unsafe region ($h_{img}(u, v) + h_d(d_s) = 0$). Therefore, the necessary CBF requirements (2) of $h(\zeta)$ automatically will be fulfilled for a more conservative safe set $S \subseteq \mathbb{X}_s$ defined in the 3D Euclidean workspace.

In summary, the system (1) can be in three different situations based on the observed image ζ as below:

- The system is in a safe set: $h(\zeta) > 0$ when $h_{img}(u, v) > 0$, or $h_{img}(u, v) < 0$ *and* $d > d_s$
- The system is on the border of unsafe set: $h(\zeta) = 0$ when $h_{img}(u, v) = 0$ *and* $d = 0$, or $h_{img}(u, v) < 0$ *and* $d = d_s$
- The system is in an unsafe set: $h(\zeta) < 0$ when $h_{img}(u, v) < 0$ *and* $d < d_s$

We are now ready to state the CBF condition in space as follows:

$$\dot{h}(\zeta) = \nabla_p h_{img}^T [\dot{u}, \dot{v}]^T + \frac{\partial h_d}{\partial d} \dot{d} \geq -\alpha(h(\zeta)) \tag{5}$$

where $\nabla_p h_{img}^T$ is the transpose of the gradient of function $h_{img}(u, v)$ in the pixel coordinate system, which can be calculated from $h_{img}(u, v)$.

Next, we will compute the relation of $\dot{u}$, $\dot{v}$, and $\dot{d}$ and actual control signals, that is, linear speed v and angular speed $\boldsymbol{\omega}$ of the robot body. Consider P_{obs} an arbitrary point on the real world, for example on an obstacle as is shown in Fig. 1. The position of obstacle expressed in camera coordinate ${}^c P_{obs}$ can be obtained as a function of $[u_{obs}, v_{obs}, d_{obs}]^T$ by a transformation: ${}^c P_{obs} = \boldsymbol{T}(u_{obs}, v_{obs}, d_{obs})$ where d_{obs} is depth value at $[u_{obs}, v_{obs}]$.

Now, we can write $[u_{obs}, v_{obs}, d_{obs}]^T$ as a function of position of the body P_{body}, position of the camera P_{cam} as below:

$${}^w P_{obs} = {}^w P_{body} + {}^w_b \boldsymbol{R}\, {}^b P_{cam} + {}^w_b \boldsymbol{R}\, {}^b_c \boldsymbol{R}\, \boldsymbol{T}(u_{obs}, v_{obs}, d_{obs}) \tag{6}$$

where ${}^w_b \boldsymbol{R}$ represents the rotation matrix from body to world coordinate ($\mathcal{F}_b \rightarrow \mathcal{F}_w$), and ${}^b_c \boldsymbol{R}$ denotes the rotation matrix from camera to body coordinate system ($\mathcal{F}_c \rightarrow \mathcal{F}_b$).

Assuming that the obstacles are stationary, the time derivative of Eq. (6) will be:

$$\begin{aligned} \vec{0} = {}^w \dot{P}_{body} + {}^w_b \dot{\boldsymbol{R}}\, {}^b P_{cam} + \\ {}^w_b \dot{\boldsymbol{R}}\, {}^b_c \boldsymbol{R}\, \boldsymbol{T}(u_{obs}, v_{obs}, d_{obs}) + \\ {}^w_b \boldsymbol{R}\, {}^b_c \boldsymbol{R}\, \dot{\boldsymbol{T}}(\dot{u}_{obs}, \dot{v}_{obs}, \dot{d}_{obs}) \end{aligned} \tag{7}$$

where ${}^w \dot{P}_{body}$ is absolute velocity of the mobile robot expressed in world coordinate, which can be written as a function of velocity magnitude and orientation of the robot: ${}^w \dot{P}_{body} = {}^w_b \boldsymbol{R}\ [\mathrm{v}, 0, 0]^T$. Furthermore, ${}^w_b \dot{\boldsymbol{R}}$ will provide a matrix which includes the angular velocity of the robot: $\boldsymbol{\omega}$. Therefore, we can write $\dot{u}$, $\dot{v}$, and $\dot{d}$ as a function of control inputs: $[\dot{u}, \dot{v}, \dot{d}]^T = F(\boldsymbol{u})$. By substituting $F(\boldsymbol{u})$ in the equation (5), we can rewrite (5) based on control inputs.

Given the control system (1), and control inputs $\boldsymbol{u} = [\mathrm{v}, \boldsymbol{\omega}]^T$ the control policy (8) will guarantee safety for any trajectory starting in S.

$$\pi_{vcbf} = \{\boldsymbol{u} \in \mathbb{U} \mid \nabla_p h_{img}^T [\dot{u}, \dot{v}]^T + \frac{\partial h_d}{\partial d} \dot{d} \geq -\alpha(h(\zeta))\} \tag{8}$$

Note that the performance of the safe controller depends on the hyperparameters of the $h_{img}(u, v)$ and $h_d(d)$, and $\alpha(.)$, which can be tuned based on the environment and mobile robot status.

This control strategy can be expanded by considering multiple points of obstacles P_{obs} throughout the image, as well as using multiple cameras to cover a wide range of robot's surrounding.

In case of using multiple V-CBF, let define $S_k = \mathbb{X} \backslash \mathbb{X}_{u_k}$ as a safe set relevant to the system (1) that is the complement of the k^{th} unsafe set ($\mathbb{X}_{u_k}$). By constructing the control barrier functions $h_k(\zeta)$ so that satisfy the required conditions in (2) for each S_k, we can define an overall safe set as intersection of individual safe sets S_k:

$$S = S_1 \cap S_2 \cap ... \cap S_{N_o} \tag{9}$$

Obviously, S satisfies the conditions on (2) for all $k \in \{1, 2, ..., N_o\}$, where N_o is the number of constructed V-CBFs. To guarantee safety, we therefore let the control policy be $\pi_{vcbf} = \pi^1_{vcbf} \cap \pi^2_{vcbf} \cap ... \cap \pi^{N_o}_{vcbf}$.

B. Control Lyapunov Function (CLF)

In addition to safety guarantee, we should define a control policy that satisfies the stability requirements. To this aim, consider a candidate Control Lyapunov Function (CLF) $V : \mathbb{X} \rightarrow \mathbb{R}$ that is continuously differentiable, and positive-definite.

Definition 6: A function $V : \mathbb{X} \rightarrow \mathbb{R}$ is said to be positive-definite in $\mathbb{X}$ if (i) $V(0) = 0$, and (ii) $V(\boldsymbol{x}) > 0$ for all $\boldsymbol{x} \in \mathbb{X}$ such that $\boldsymbol{x} \neq 0$.

The necessary and sufficient condition that guarantees the stability of (1) in equilibrium point, is [21]:

$$\inf_{\boldsymbol{u} \in \mathbb{U}} [\dot{V}(\boldsymbol{x}, \boldsymbol{u}) = L_f V(\boldsymbol{x}) + L_g V(\boldsymbol{x}) \boldsymbol{u}] \leq -\gamma(V(\boldsymbol{x})) \tag{10}$$

where $\gamma(.)$ is an extended class κ function. Therefore, the following control policy will satisfy the stability requirements:

$$\pi_{clf} = \{\boldsymbol{u} \in \mathbb{U} \mid \dot{V}(\boldsymbol{x}, \boldsymbol{u}) \leq -\gamma(V(\boldsymbol{x}))\} \tag{11}$$

C. Integrating CLF and V-CBF

To satisfy both safety and stability requirements, the proposed control policy should be the intersection of two previous policies, that is, $\pi^* = \pi_{vcbf} \cap \pi_{clf}$. As we will see below, when the intersection is empty, we need to prioritise the safety. Now, we can define a constrained optimisation problem to find an optimum control input that fulfils both safety and stability constraints simultaneously:

$$\boldsymbol{u}^* = \underset{\boldsymbol{u}}{\operatorname{argmin}}\ J(\boldsymbol{u}, \delta) \tag{12a}$$

$$\text{s.t.}: \quad \dot{V}(\boldsymbol{x}, \boldsymbol{u}) \leq -\gamma(V(\boldsymbol{x})) + \delta \tag{12b}$$

$$\nabla_p h^T_{img} [\dot{u}_k, \dot{v}_k]^T + \frac{\partial h_d}{\partial d} \dot{d}_k \geq -\alpha(h_k(\zeta))\ k = 0, \ldots, N_o \tag{12c}$$

$$\boldsymbol{u}_{lb} \leq \boldsymbol{u} \leq \boldsymbol{u}_{ub} \tag{12d}$$

where J denotes the cost function, $\boldsymbol{u}_{lb}$ and $\boldsymbol{u}_{ub}$ are control input's lower and upper bound. Moreover, for sake of clarity, we have not replaced $[\dot{u}_k, \dot{v}_k, \dot{d}_k]^T = F_k(\boldsymbol{u})$ in the above equations. Furthermore, δ denotes the relaxation parameter that will be activated when the constraints conflict with each other and then the intersection $\pi_{vcbf} \cap \pi_{clf}$ is empty. In this case, the hard constraint (safety) will be preserved, and the soft constraint (stability) will be softened.

In the Algorithm (1) steps of deployment of the vision-based safe control is stated.

Algorithm 1: Vision-based Safety Control Algorithm

Input: RGB-D image: ζ, States: $\boldsymbol{x}$
Output: Control Input: $\boldsymbol{u}$

```
while Robot is running do
    receive image ζ
    generate h_img(u, v), and ∇_p h_img(u, v)
    calculate h_d(d), and ∂h_d/∂d
    run optimization (12) to obtain control inputs: u*
```

Next, we will present how we train functions $h_{img}(u, v)$, and $\nabla_p h_{img}(u, v)$.

IV. Contour Generation

Based on the discussion in the previous section, here we aim to generate the map $h_{img}(u, v)$ with the RGB-D image as input to create the barrier function $h(\zeta)$. To this aim, several image processing techniques, like distance transform, or learning-based methods, such as image-to-image translation, can be adopted. Fig. 2 illustrates our computational pipeline, which takes the current RGB-D image as input. At first, the safe and unsafe areas are segmented by an instance segmentation algorithm. Then, the far obstacles are removed from the segmented image using a threshold distance and the depth image. After the explained pre-process, the result will go through a h_{img}-generator unit to produce 2D image $h_{img}(u, v)$. Now, the 2D gradient field can be created by applying the gradient operator ∇_p on the provided $h_{img}(u, v)$.

The core of our vision-based control barrier function scheme (the generator unit) is an image-to-image translation problem. Here, we aim to provide a learning-based mapping from preprocessed RGB-D image to the image $h_{img}(u, v)$. Consider the following image-to-image map:

$$h_{img} = \Gamma_{\boldsymbol{w}}(\zeta') \tag{13}$$

where $\boldsymbol{w}$ denotes the mapping parameters, and $\zeta' \in \mathbb{R}^{W \times H \times 4}$ is the preprocessed RGB-D image ζ, and function $h_{img} \in \mathbb{R}^{W \times H}$ represent the values of function h_{img} in each pixel (u, v). In this study $W = 255$ and $H = 255$. The image to image translation has been addressed in various studies: U-Net [22], cGAN [23], SegNet [24], CycleGAN [25], StarGAN [26]. In this study, the conditional Generative Adversarial Network (cGAN), which is a commonly used image-to-image translation network, is employed.

A. Conditional Generative Adversarial Network (cGAN)

To train the cGAN, consider the following loss function [23]:

$$\begin{aligned} \mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{\zeta_i, \zeta_o}[\log D(\zeta_i, \zeta_o)] + \\ \mathbb{E}_{\zeta_i, \boldsymbol{\xi}}[\log(1 - D(\zeta_i, G(\zeta_i, \boldsymbol{\xi})))] \end{aligned} \tag{14}$$

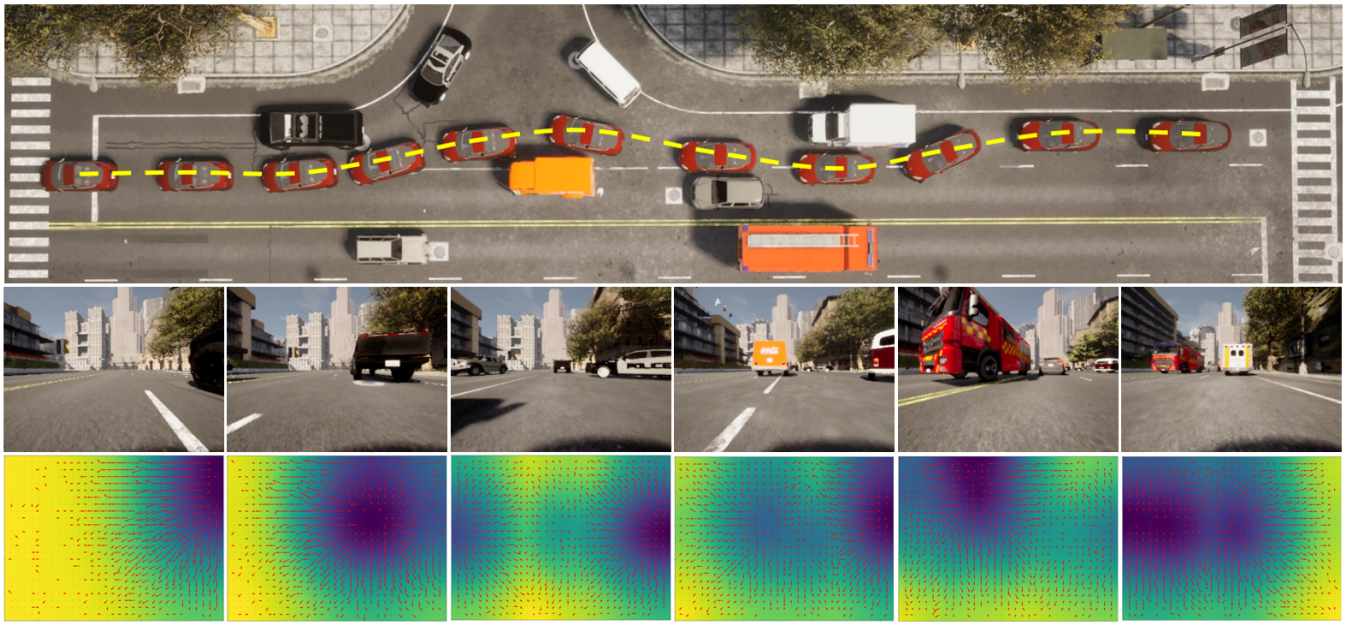

Fig. 3. Top-view snapshot of simulation in CARLA environment. The red car is an ego self-driving vehicle using V-CBF to navigate safely. The yellow dashed line is the path of the autonomous car. The second row shows the front view of the camera installed on the car, and the last line illustrates the generated contour of $h_{img}(u,v)$ and $\nabla_p h_{img}(u,v)$

where ζ_i is the input image (preprocessed RGB-D image), ζ_o denotes the output image (h_{img}), and $\boldsymbol{\xi}$ indicates the random noise vector. Also, the generator $G : \{\zeta_i, \boldsymbol{\xi}\} \rightarrow \zeta_o$ map the observed image ζ_i and random noise vector $\boldsymbol{\xi}$ to the output image ζ_o, and D is the discriminator network. We aim to train the map $\Gamma_{\boldsymbol{w}}(.)$ so that the G minimizes the loss function and D maximizes it. Therefore, the final trained model will be:

$$\boldsymbol{G^*} = \arg\min_G \max_D \Big(\mathcal{L}_{cGAN}(G,D) + \lambda_1 \mathcal{L}_h(G) + \lambda_2 \mathcal{L}_{\nabla h}(G) \Big) \tag{15}$$

where $\mathcal{L}_h(G) = \mathbb{E}_{\zeta_i,\zeta_o,\boldsymbol{\xi}}[\|\zeta_o - G(\zeta_i,\boldsymbol{\xi})\|_1]$, and $\mathcal{L}_{\nabla h}(G) = \mathbb{E}_{\zeta_i,\zeta_o,\boldsymbol{\xi}}[\|\partial\zeta_o/\partial u - \partial G(\zeta_i,\boldsymbol{\xi})/\partial u\|_1 + \|\partial\zeta_o/\partial v - \partial G(\zeta_i,\boldsymbol{\xi})/\partial v\|_1]$ to train the map near the ground truth of ζ_o, $\partial\zeta_o/\partial u$, and $\partial\zeta_o/\partial v$. Furthermore, λ_1 and λ_2 denote the regularization weights.

B. Ground Truth and Training Data-set

This section will describe steps of generating ground truth through visual image processing techniques. The objective is to build a ground truth so that h_{img} satisfies the necessary properties as described earlier.

In the first step, the RGB-D image ζ is segmented into safe and unsafe regions. Safe areas are defined as pixels representing the spaces in the environment with which the robot cannot have contact or its collision is not destructive. For instance, in an autonomous car, roads, leaves of trees, and the sky can be considered safe areas, and the other zones will be labeled as unsafe. The benefit of this method is that the model only should be trained to identify the areas in the safe category which are limited instead of recognizing all unlimited unsafe areas.

In the next step, based on the depth image, the unsafe areas farther than a threshold value (d^*) are removed to prioritize the close objects compared to the far ones. For the remained unsafe zones, we apply an instance segmentation to create m instances of binary mask images $\zeta'_j, j = 1, ...m$.

Then, by applying a two-dimension Gaussian smoothing blur filter on the binary mask images and remapping them, a smooth function for each obstacle will be produced so that it continuously increases from negative values inside the unsafe zone to positive values outside them, and is equal to zero on the border of the unsafe areas.

Based on the discussion in section III-A, to restrict the values of function h_{img} between a lower and upper bound, a bounded activation function is applied on the results.

We finally combine all the filtered binary masks to a single image by $\zeta'' = \min_j \zeta''_j, \forall(u,v)$. Recall that the image values are negative inside the unsafe region. Thus, the minimization takes the "most unsafe" pixle into account. Finally, a two-dimension Gaussian smoothing blur filter will be applied to keep the generated ζ'' a smooth continuously differentiable function, and adjust the sharpness of the gradient of ζ''. The aforementioned steps for providing ground truth have been shown in the Algorithm (2).

Before closing this section, we will show that in fact the Gaussian filter guarantees that the resulting images are differentiable and their gradients can be made arbitrary small. Consider a unit step function $H(x)$ that represents an edge in a binary image. Applying a Gaussian filter on it will result:

$$(Gau * H)(x) = \int_{-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-(\tau-\mu)^2/2\sigma^2} H(x-\tau) d\tau \tag{16}$$

where Gau is Gaussian function, μ and σ are mean and

Algorithm 2: Ground Truth generation

Input: RGB-D image: ζ
Output: Ground Truth: ζ_o, $\partial\zeta_o/\partial u$, $\partial\zeta_o/\partial v$

segment the image ζ into safe and unsafe regions.
remove the unsafe areas farther than d^*.
do instance segmentation for near unsafe zones.
$\zeta'_j \leftarrow$ create a binary mask image for each instance.
for *all* ζ'_j **do**
    $Gau(\zeta'_j) \leftarrow$ apply Gaussian filter on ζ'_j
    remap $Gau(\zeta'_j)$ to satisfy CBF boundary property
    $\zeta''_j \leftarrow$ bound the results.
for *all pixels* (u, v) **do**
    $\zeta''(u, v) \leftarrow min(\zeta''_j(u, v))$
apply a Gaussian filter to remove any discontinuity: $\zeta_o \leftarrow Gau(\zeta'')$
calculate: $\partial\zeta_o/\partial u$, $\partial\zeta_o/\partial v$

standard deviation, respectively. It can be mathematically proven that the derivative of blurred image at an arbitrary edge will be as below:

$$D_x(Gau * H) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{17}$$

where D_x denotes the derivative in the direction of x, whose maximum value is $max(D_x(Gau * H)) = \frac{1}{\sigma\sqrt{2\pi}}$. By choosing a high-enough standard deviation σ^* for the Gaussian filter, it can be guaranteed that the gradient of the blurred image will be bounded as below:

$$0 < D_x(Gau * H) \leq \frac{1}{\sigma^*\sqrt{2\pi}} \tag{18}$$

Therefore, the ground truth of $\partial\zeta_o/\partial u$, $\partial\zeta_o/\partial v$ remains bounded, and the contour of ζ_o would be smooth and continuously differentiable.

V. Validation

In this section, we will demonstrate the efficacy of the introduced control strategy on a self-driving car in CARLA environment, an urban driving simulator [27]. To this end first we collected a data-set containing 3700 RGB-D images from CARLA environment. The network is trained using ground truth generated by Algorithm 2.

In this study, the Adam optimizer with a learning rate of 0.0002 is utilized for training the cGAN. The quantitative evidence of training indicates that with 150 epochs and batch size equal to 10, in the validation scenario, the cGAN reaches 0.11 mean absolute error for h_{img} and 0.006 mean absolute error for $\nabla_p h_{img}$, where the input images are normalized between -1 and 1.

Considering the linear and angular velocity as control inputs $\boldsymbol{u} = [\mathrm{v}, \omega]^T$, the steering and throttle commands can be calculated based on the standard kinematic bicycle model in a 2D workspace [28]. Furthermore, we attached three cameras to the ego vehicle to cover the front, left, and right sides of the car. The simulation is conducted in an unknown environment with various stationary cars as obstacles. In addition, a quadratic Lyapunov function is employed to keep the heading equal to desired value, and the cost function of the optimization is created in a way that maintains the car speed at the desired value $\mathrm{v}_d = 3m/s$.

The Fig. 3, qualitatively demonstrates that V-CBF controller in fact navigates the car safely in an unknown environment with stationary obstacles. In this Figure, the yellow dashed line shows the path of the ego vehicle. The second row shows the front view of the camera installed on the autonomous car, and the last line displays h_{img} and its gradient field generated by cGAN.

Fig. 4 illustrates the control input $\boldsymbol{u} = [\mathrm{v}, \omega]^T$ generated by Algorithm 1 (solid line), and the actual values of the linear and angular velocity of the vehicle (dashed line) with respect to the longitudinal position of the car on the road. Note that in Fig. 3 the car starts from right side of the page and travel to the left. Similarly, the command plots in 4 start from the right.

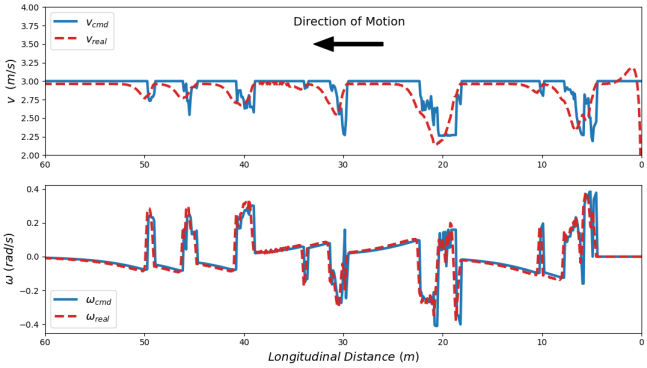

Fig. 4. Linear (up) and angular velocity (down) of the autonomous vehicle vs. longitudinal distance on the road for the values of command (solid line) and real measurement (dashed line).

VI. Conclusion

This paper proposes a vision-based method for constructing a control barrier function (CBF) in real-time using a stream of RGB-D images from a camera as an input to a conditional Generative Adversarial Network (cGAN) for image segmentation and image-to-image translation. The method can be used to calculate safety conditions for mobile robots navigating in unknown environments and arbitrary obstacles. The proposed method was demonstrated in the CARLA simulator, where a car could navigate safely in a previously unknown environment. Our method also allowed us to use three cameras facing forward and sides to increase the field of view. Although the introduced V-CBF concept is general, part of the derivations is only presented for wheeled mobile robots. In the future, we will demonstrate the performance of the proposed V-CBF on real vehicles in a dynamic environment, and investigate the suitability of other models and methods for learning the CBF, for example, is it better to be end-to-end or modular.

REFERENCES

[1] G.-Z. Yang, J. Bellingham, P. E. Dupont, P. Fischer, L. Floridi, R. Full, N. Jacobstein, V. Kumar, M. McNutt, R. Merrifield, *et al.*, "The grand challenges of science robotics," *Science robotics*, vol. 3, no. 14, p. eaar7650, 2018.

[2] W. Xiao, T.-H. Wang, M. Chahine, A. Amini, R. Hasani, and D. Rus, "Differentiable control barrier functions for vision-based end-to-end autonomous driving," *arXiv preprint arXiv:2203.02401*, 2022.

[3] Y. Meng, Z. Qin, and C. Fan, "Reactive and safe road user simulations using neural barrier certificates," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6299–6306, IEEE, 2021.

[4] P. Glotfelter, J. Cortés, and M. Egerstedt, "Nonsmooth barrier functions with applications to multi-robot systems," *IEEE control systems letters*, vol. 1, no. 2, pp. 310–315, 2017.

[5] Y. Chen, A. Singletary, and A. D. Ames, "Guaranteed obstacle avoidance for multi-robot operations with limited actuation: A control barrier function approach," *IEEE Control Systems Letters*, vol. 5, no. 1, pp. 127–132, 2020.

[6] B. Xu and K. Sreenath, "Safe teleoperation of dynamic uavs through control barrier functions," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7848–7855, IEEE, 2018.

[7] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2460–2465, IEEE, 2018.

[8] Q. Nguyen, A. Hereid, J. W. Grizzle, A. D. Ames, and K. Sreenath, "3d dynamic walking on stepping stones with control barrier functions," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 827–834, IEEE, 2016.

[9] R. Grandia, A. J. Taylor, A. D. Ames, and M. Hutter, "Multi-layered safety for legged robots via control barrier functions and model predictive control," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8352–8358, IEEE, 2021.

[10] Y. Huang, M. Zhu, Z. Zheng, and K. H. Low, "Linear velocity-free visual servoing control for unmanned helicopter landing on a ship with visibility constraint," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 5, pp. 2979–2993, 2021.

[11] D. Zheng, H. Wang, J. Wang, X. Zhang, and W. Chen, "Toward visibility guaranteed visual servoing control of quadrotor uavs," *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 3, pp. 1087–1095, 2019.

[12] H.-A. Hung, H.-H. Hsu, and T.-H. Cheng, "Image-based multi-uav tracking system in a cluttered environment," *IEEE Transactions on Control of Network Systems*, 2022.

[13] C. Dawson, S. Gao, and C. Fan, "Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods," *arXiv preprint arXiv:2202.11762*, 2022.

[14] L. Wang, D. Han, and M. Egerstedt, "Permissive barrier certificates for safe stabilization using sum-of-squares," in *2018 Annual American Control Conference (ACC)*, pp. 585–590, IEEE, 2018.

[15] W. Xiao, R. Hasani, X. Li, and D. Rus, "Barriernet: A safety-guaranteed layer for neural networks," *arXiv preprint arXiv:2111.11277*, 2021.

[16] R. K. Cosner, Y. Yue, and A. D. Ames, "End-to-end imitation learning with safety guarantees using control barrier functions,"

[17] C. Dawson, B. Lowenkamp, D. Goff, and C. Fan, "Learning safe, generalizable perception-based hybrid control with certificates," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1904–1911, 2022.

[18] K. Long, C. Qian, J. Cortés, and N. Atanasov, "Learning barrier functions with memory for robust safe navigation," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4931–4938, 2021.

[19] M. Srinivasan, A. Dabholkar, S. Coogan, and P. A. Vela, "Synthesis of control barrier functions using a supervised machine learning approach," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7139–7145, IEEE, 2020.

[20] F. Blanchini, "Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[21] 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, IEEE, 2019.

[22] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[23] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

[24] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[25] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

[26] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8789–8797, 2018.

[27] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*, pp. 1–16, PMLR, 2017.

[28] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.