Counteracting Adversarial Attacks in Autonomous Driving

Qi Sun, *Graduate Student Member, IEEE*, Xufeng Yao, Arjun Ashok Rao, Bei Yu, *Member, IEEE*, and Shiyan Hu, *Senior Member, IEEE*

***Abstract*—This article studies the robust deep stereo vision in autonomous driving systems and counteracting adversarial attacks. The autonomous system operation requires real-time processing of measurement data which often contain significant uncertainties and noise. Adversarial attacks have been widely studied to simulate these perturbations in recent years. To counteract the practical attacks in autonomous systems, novel methods based on simulated attacks are proposed in this article. Univariate and multivariate functions are adopted to represent the relationships between the left and right input images and the deep stereo model. A stereo regularizer is proposed to guide the model to learn the implicit relationship between the images and characterize the loss function's local smoothness. The attacks are generated by maximizing the regularizer term to break the linearity and smoothness. The model then defends the attacks by minimizing the loss and regularization terms. Two techniques are developed in this article. The first technique, `SmoothStereo`, explores the basic knowledge from the physical world and smoothness, while the second technique, `SmoothStereoV2`, improves `SmoothStereo` through leveraging the smooth activation functions during the defense. `SmoothStereoV2` can learn and utilize the gradient information concerning the attacks. The gradients of the smooth activation functions can handle attacks for improving the model robustness. Numerical experiments on KITTI datasets demonstrate that the proposed methods offer superior performance.**

***Index Terms*—Adversarial defense, autonomous system, local smoothness, robust stereo vision.**

I. Introduction

With the arrival of the artificial intelligence era, autonomous driving systems based on deep neural networks (DNNs) have triggered a new revolution in traveling and have a high potential to change the development of cities. An autonomous driving system needs to complete the following tasks: sensing, decision making, planning, and control. Among these, sensing is considered the most fundamental and important task. In recent years, vision-based matching and detection systems that utilize DNNs have been widely used as sensing systems [2].

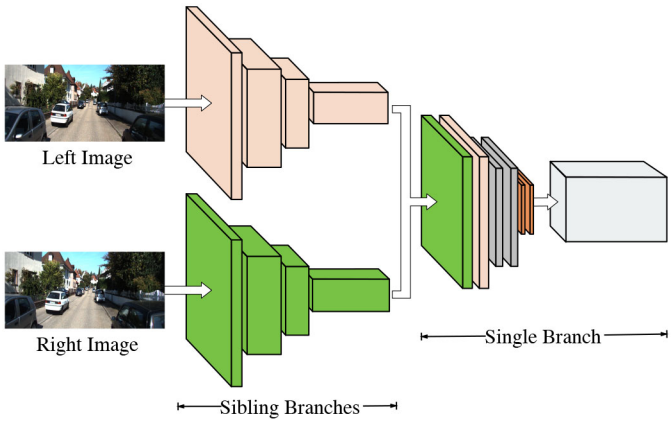


Fig. 1. Structure of a typical stereo-based deep vision model. There are two sibling branches, e.g., feature extraction modules and RPN modules. Each branch takes left and right images as input, respectively. The extracted feature maps or object proposals are concatenated or reshaped into a single feature map for further processing, e.g., regressing 3-D boundary cube, predicting viewpoints, generating disparity maps, etc. Some models may concatenate more branches after the sibling branches to conduct complicated tasks, relying on the knowledge from these sibling branches.

A stereo-based model is a vision-based system that exploits sparse, dense, semantic, or geometrical information in stereo imagery. Most of these models, e.g., Faster R-CNN [3], utilize large feature networks as their backbone to extract features and use region proposal networks (RPNs) to generate object proposals which are then refined in subsequent modules. With this rich information, we can get more accurate key points, viewpoints, object dimensions, disparity maps, bounding boxes of the 3-D objects, etc [4]–[10]. Usually, the left and the right images cooperate in the stereo-vision system, as shown in Fig. 1. Three-dimensional spatial knowledge is highly dependent on the left and right stereo-pair images. Contrary to the stereo systems, monocular approaches suffer from the lack of accurate depth information and, as a result, cannot provide comparable performance [8].

Although deep learning algorithms have demonstrated superior performance in many circumstances, recent researches reveal that these algorithms are vulnerable to perturbations. This security risk is hazardous for stereo models used in

Manuscript received 21 September 2021; revised 8 February 2022; accepted 13 March 2022. Date of publication 8 April 2022; date of current version 22 November 2022. This work was supported in part by SmartMore; in part by the ITF Partnership Research Programme under Grant PRP/65/20FX; and in part by the Research Grants Council of Hong Kong under Project CUHK14209420. The preliminary version [1] has been presented at the IEEE/ACM International Conference on Computer-Aided Design (ICCAD) in 2020 [DOI: 10.1145/3400302.3415758]. This article was recommended by Associate Editor R. S. Chakraborty. *(Corresponding author: Bei Yu.)*

Qi Sun, Xufeng Yao, Arjun Ashok Rao, and Bei Yu are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, SAR (e-mail: byu@cse.cuhk.edu.hk).

Shiyan Hu is with the Department of Computer Science and Electronic Engineering, University of Southampton, Southampton SO17 1BJ, U.K.

Digital Object Identifier 10.1109/TCAD.2022.3166112

autonomous driving systems. Consequently, the concept of adversarial attacks [11] came into being to measure these perturbations. Typically, adversarial perturbations are crafted to be invisible to human observers and indistinguishable from the original image. This is achieved by constraining the ℓ_p norm of the adversarial image to a predefined value that ensures human imperceptibility from a pixel-difference perspective. However, when added to images, adversarial examples cause significant errors in the stereo model. Several adversarial attack algorithms have been designed to attack DNN models [11]–[19]. Szegedy *et al.* [11] first demonstrated the existence of perturbations to natural images that can fool DNN models into misclassification. To generate adversarial images more efficiently, Goodfellow *et al.* [12] proposed a novel method termed the fast gradient sign method (FGSM) to generate the perturbations by computing the gradient of the loss function. Intuitively, this method updates each input image pixel through its gradient to maximize the loss while model parameters are kept unchanged. FGSM utilizes the linearity hypothesis of DNN models, i.e., designs of deep learning models encourage linear behavior for computational gains. The basic iterative method (BIM, also known as, I-FGSM) [16] extended FGSM by iteratively taking multiple small steps to adjust the perturbation direction. Projected gradient descent (PGD) [17] studied the adversarial perturbations from the perspective of optimization. PGD initializes the search for an adversarial image at a random point within the perturbation range.

Attacking the stereo models is more challenging compared to attacking the classification models. Typically, a classification model comprises a feature extraction module and a classifier. In contrast, the stereo models are composed of many complicated modules to learn enough knowledge, which would contain lots of redundant information. The classification tasks target the class labels in one-hot encodings, while the stereo models detect vehicles that are more difficult. Chen *et al.* [20] attacked detectors via the expectation over transformation (EOT) technique—a method that computes the perturbation by adding random distortions (e.g., resizing, rotation, etc.) to natural images. Li *et al.* [18] attacked the shapes of bounding boxes and classification labels simultaneously. Li *et al.* [19] and Dong *et al.* [21] attacked more relevant objects by splitting the whole image into subregions, e.g., foreground and background, or several superpixels. Adversarial examples also exist in the physical world. Some adversarial images and road signs are printed to fool deep vision models [16], [22]. Adversarial T-shirts can deceive detection systems with a few adversarial patches on the clothing [23], [24]. Athalye *et al.* [25] generated adversarial 3-D objects via transformation-based methods.

Some adversarial defense algorithms have been proposed to improve model robustness to attacks [26], [27]. A majority of the literature introducing new adversarial attack methods trained the models with their attacked inputs which is a practice termed adversarial training [12], [16], [17]. Some methods modified the raw inputs by conducting preprocessing operations, e.g., random resizing [28] and data compression [29] to introduce the perturbations to the inputs artificially and help the model learn the critical information robustly under the perturbations. SafetyNet [30] proposed to append an SVM classifier to the models such that SVM can use the discrete feature codes. For an input image, its discrete codes are compared against the codes of training data to determine whether it is an adversarial image. Generative adversarial networks (GANs) [31], [32], composed of generators and discriminators, add two novel modules to help generate perturbations and discriminate adversarial inputs. Stereopagnosia [33] discusses the influence of imposing traditional attack methods directly on the stereo models and shows that the stereo models are vulnerable to attacks. However, there has been no specifically designed work done on defending against attacks on stereo-based models to the best of our knowledge.

In this article, we propose a novel attack method with a physically meaningful regularization term that considers the characteristics of the stereo models and use a smooth defense method based on adversarial training to tune the model. Stereo-based models usually utilize the implicit spatial information from the left and right images to extract features independently, i.e., the sibling branches in Fig. 1. The concatenated features from these two images are further fused to learn more information jointly, i.e., the single branch in Fig. 1. Considering that these two types of mechanisms can be modeled as univariate and multivariate functions, a novel stereo-based regularizer targeting the overall loss of these functions is proposed. The remainders of Taylor expansions represent the regularizer term to characterize the local linearity and smoothness of the loss surface. Our attacks tend to break the smoothness of the loss surface by maximizing the regularizer term to generate the perturbations. The direct smooth defenses minimize the loss concerning the perturbations to improve the smoothness. With these features, our novel defense method can counteract adversarial attacks efficiently. The proposed method is named `SmoothStereo` [1].

We further improve our method to be `SmoothStereoV2` by using the smooth activation function during the model defense. Some activation functions are employed, including SoftPlus [34], exponential linear unit (ELU) [35], Gaussian error linear unit (GELU) [36], and Swish [37], in place of the widely used nonsmooth activation function rectified linear unit (ReLU) in existing stereo models. Compared with the direct defense method in `SmoothStereo`, the smooth activation function can capture and utilize more feature and gradient information concerning the attack images during defense training and introduce more smoothness into the model without changing the model structures. Therefore, using the smooth activation function ameliorates the loss function during training, and the model is trained to be more robust to perturbations. Furthermore, using smooth activation functions will not cause extra inference costs because of no changes to the model structures and tiny computation workloads of activation functions. The importance and effectiveness of the smoothness have been emphasized in some previous arts [38]–[42].

Focusing on the features and gradients is always of vital importance to improve the robustness of the stereo models, and how to handle these has naturally been a critical topic of the adversarial defenses no matter whether it is for stereo applications or not [12], [16], [17], [26]–[28], [43].

Studying the gradients is also a key topic of various applications and theoretical problems. Akiyama and Suzuki [44] discussed the two-layer ReLU used in teacher–student learning and Asi *et al.* [45] proposed an adaptive gradient method in convex optimization. Saito *et al.* [46], Jiang *et al.* [47], Sitawarin *et al.* [48], and Cui *et al.* [49] focused on different tasks with distinct settings from our stereo applications and designed new loss functions to manipulate the gradients and features in their problems.

The experimental results on the KITTI 2012 and KITTI 2015 stereo datasets [50], [51] show the outstanding performance of our stereo-based perturbation generation method and smooth defense method under various strengths of attacks, compared with FGSM [12], PGD [17], and I-FGSM [16].

The remainder of this article is organized as follows. Section II introduces the problem to be addressed and preliminaries. Section III explains our proposed method `SmoothStereo` in detail. Section IV proposes the novel and more powerful `SmoothStereoV2` based on smooth activation functions. Section V summarizes the defense flow. Section VI demonstrates the experiments, the results, and analyses. Finally, Section VII concludes this article.

II. Preliminaries

A. Adversarial Training

Adversarial training can be traced back to the rise of adversarial attack algorithms [12]. The typical form of the most representative adversarial training algorithms involves two steps: 1) the generation of the adversarial image set via *adversarial attacks* and 2) the *defense training* based on the adversarial images. Most adversarial training methods perform the following min–max training strategy shown in:

$$\begin{aligned} \arg\min_{\boldsymbol{\theta}} \ & \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in\mathbb{D}}\left[\max_{\boldsymbol{\delta}\in\Delta} L(\boldsymbol{x}+\boldsymbol{\delta},\boldsymbol{\theta};\boldsymbol{y})\right] \\ \text{s.t.} \ & \|\boldsymbol{\delta}\|_p \le \boldsymbol{\epsilon} \end{aligned} \tag{1}$$

where $\mathbb{D}$ represents the dataset with $\boldsymbol{x}$ as the input image, $\boldsymbol{y}$ as the ground truth, $\boldsymbol{\theta}$ represents the model parameters, $\boldsymbol{\delta}$ denotes the perturbations, Δ is the perturbation set corresponding to $\mathbb{D}$, $L(\cdot)$ is the loss function, and $\mathbb{E}(\cdot)$ represents the expected loss over $\mathbb{D}$. $\|\cdot\|_p$ is the ℓ_p-norm, which constrains the perturbation within $\boldsymbol{\epsilon}$ such that the perturbation is imperceptible. $\boldsymbol{\epsilon}$ reflects the strengths of the attacks. The larger $\boldsymbol{\epsilon}$ permits a more extensive range of perturbations. The generated images deviate farther from the clean images and, therefore, are more harmful to the deep learning models. In comparison, a smaller $\boldsymbol{\epsilon}$ leads to weaker perturbations. Some experimental results in Table IV can be taken as examples to illustrate this. For simplicity, we use Δ to represent the candidate perturbation set which is updated under the constraint $\boldsymbol{\epsilon}$. First, in (1), we maximize the model loss to learn the adversarial perturbation $\boldsymbol{\delta}$ for each image $\boldsymbol{x}$ via gradient ascent

$$\begin{aligned} \boldsymbol{\delta}_{i+1} &= \boldsymbol{\delta}_i + \boldsymbol{s}\cdot\nabla_{\boldsymbol{\delta}_i} L(\boldsymbol{x}+\boldsymbol{\delta}_i,\boldsymbol{\theta};\boldsymbol{y}) \\ \boldsymbol{\delta}_{i+1} &= \text{clamp}(\boldsymbol{\delta}_{i+1};-\boldsymbol{\epsilon},\boldsymbol{\epsilon}) \end{aligned} \tag{2}$$

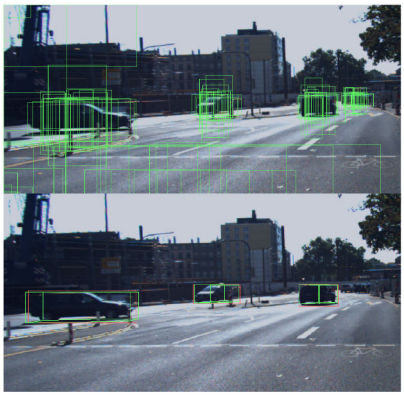


Fig. 2. Generated object proposals and the final detected objects. The proposals determine the final detection objects.

where $\boldsymbol{s}$ is the step size and i is the optimization step with

$$\boldsymbol{\delta}_0 = \text{random}(-\boldsymbol{\epsilon},\boldsymbol{\epsilon}) \tag{3}$$

where random$(\cdot)$ is to generate the random initial perturbation in $[-\boldsymbol{\epsilon},\boldsymbol{\epsilon}]$, and clamp$(\cdot)$ is the clamping function to force $\|\boldsymbol{\delta}_i\|_p$ to fall into the perturbation range. Sometimes researchers constrain $\boldsymbol{\delta}_i \in [-\boldsymbol{\epsilon},\boldsymbol{\epsilon}]$. The input images with their corresponding perturbations constitute the adversarial set. Second, the adversarial set is used to tune the model parameters to minimize the model loss via gradient descent

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \boldsymbol{\eta}\cdot\nabla_{\boldsymbol{\theta}_i} L(\boldsymbol{x}+\boldsymbol{\delta},\boldsymbol{\theta}_i;\boldsymbol{y}) \tag{4}$$

where $\boldsymbol{\eta}$ is the step size. $\boldsymbol{\delta}$ is obtained by solving (2). The initial parameter $\boldsymbol{\theta}_{i=0}$ is obtained from the pretrained model. After optimization via (4), the model with the updated parameters is the robust model to counteract the adversarial attacks.

B. Deep Stereo Models

Deep stereo models have proved successes in autonomous driving systems [6]–[10]. They are utilized to perform the stereo matching, object detection, disparity prediction, and regression tasks by exploiting semantic and geometric information in stereo imagery. The network architecture can be briefly divided into two parts, as shown in Fig. 1. The first module contains two sibling branches that independently extract features or generate object bounding proposals for the left and right images. The subsequent module fuses the sibling features and generates the boundary cube, key point, disparity maps, and other related 3-D spatial information. An example is shown in Fig. 2.

C. Problem Formulation

Denote $\boldsymbol{x}_l$ and $\boldsymbol{x}_r$ as the input left and right images, respectively. The ground-truth features learned from the left and right images are $\boldsymbol{b}_l$ and $\boldsymbol{b}_r$, e.g., the object proposals or the disparity features. The model prediction objective is y, e.g., the object box and disparity map. Given a stereo-based model with parameters $\boldsymbol{\theta}$ and loss function L, our task is to solve the following min–max problem:

$$\arg\min_{\boldsymbol{\theta}} \ \mathbb{E}_{(\boldsymbol{x}_l,\boldsymbol{x}_r,\boldsymbol{y})\in\mathbb{D}}\left[\max_{\boldsymbol{\delta}_l,\boldsymbol{\delta}_r\in\Delta} L_o(\boldsymbol{x}_l+\boldsymbol{\delta}_l,\boldsymbol{x}_r+\boldsymbol{\delta}_r,\boldsymbol{\theta};\boldsymbol{b}_l,\boldsymbol{b}_r,\boldsymbol{y})\right]$$

Fig. 3. Bounding boxes generated for the left and right images.

$$\text{s.t.} \quad \|\boldsymbol{\delta}_l\|_p \leq \boldsymbol{\epsilon}, \|\boldsymbol{\delta}_r\|_p \leq \boldsymbol{\epsilon} \tag{5}$$

where $\boldsymbol{\delta}_l$ and $\boldsymbol{\delta}_r$ represent the perturbations on the left and right images, respectively, and are both constrained within the perturbation budget $\boldsymbol{\epsilon}$. In this article, we propose new techniques by manipulating the loss function to counteract the attacks. For clarity, we denote the original loss function in (5) as L_o. The optimization process of (5) follows (2) and (4), while the specific form of the loss function is adopted to improve the performance.

III. Smooth Adversarial Stereo Method

In this section, we propose the `SmoothStereo` method, composed of the smoothness-driven generation of adversarial attacks and the corresponding smooth defense based on these attacks.

A. Stereo-Based Regularizer

The stereo-based deep models can handle various tasks while different tasks can be modeled as distinctive forms of objective functions. For example, the sibling RPN modules generate bounding boxes for the left and right images, respectively (as shown in Fig. 3). Therefore, we can model this part as two independent univariate functions. The regularization term should constrain both of these two functions. Regressing the 3-D bounding box or generating the final disparity map can be represented as a multivariate function. The features learned from the left–right stereo pair are jointly used as inputs to the multivariate function. Consequently, the regularization term should also be able to handle multivariate functions. We should consider both of these function terms in the regularizer to characterize the local smoothness of the loss surface.

The features learned from the left and right images share high intersections, e.g., the intersection over union (IoU) of the two regressed bounding boxes in Fig. 3. This phenomenon is consistent with the realistic understanding that stereo cameras capture the same field of view from a rectified stereo pair with a small level of disparity. However, the two features contain differences influenced by physical factors, such as the distance between the car and the object, the object's orientation to the stereo camera, etc. These physical factors vary with environments, making them expensive to measure accurately. For simplicity, we compute the distance between the two features to characterize the effects of the realistic physical factors.

Let $f_l(\boldsymbol{x}_l)$ and $f_r(\boldsymbol{x}_r)$ denote two univariate functions, to represent the features extracted from the left image $\boldsymbol{x}_l$ and right image $\boldsymbol{x}_r$, respectively. Therefore, the distance between $f_l(\boldsymbol{x}_l)$ and $f_r(\boldsymbol{x}_r)$ is defined as follows:

$$d(\boldsymbol{x}_l, \boldsymbol{x}_r) = \|f_l(\boldsymbol{x}_l) - f_r(\boldsymbol{x}_r)\|_n \tag{6}$$

where $\|\cdot\|_n$ is the ℓ_n norm. As mentioned above, the physical characteristics are measured with $d(\boldsymbol{x}_l, \boldsymbol{x}_r)$. After attacking the images, the corresponding distance is computed as follows:

$$d(\boldsymbol{x}_l + \boldsymbol{\delta}_l, \boldsymbol{x}_r + \boldsymbol{\delta}_r) = \|f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r)\|_n. \tag{7}$$

The loss term for the sibling branches under attacks is defined as follows:

$$L_b = \|d(\boldsymbol{x}_l + \boldsymbol{\delta}_l, \boldsymbol{x}_r + \boldsymbol{\delta}_r) - d(\boldsymbol{x}_l, \boldsymbol{x}_r)\|_n. \tag{8}$$

To improve the robustness of the detection system, we will minimize (8) to teach the model to reserve the physical characteristics under attacks. However, minimizing (8) would possibly result in inflexible optimization and ambiguous convergence status [52]. The specific hazard is that pushing $d(\boldsymbol{x}_l, \boldsymbol{x}_r)$ close to zero makes the model confuse the left and right images. In other words, the optimization process forces the left and right branches to output the same results, which contradicts our goal. For example, $d(\boldsymbol{x}_l, \boldsymbol{x}_r) = 0$ would result in $f_l(\boldsymbol{x}_l) = f_r(\boldsymbol{x}_r)$. So is for $d(\boldsymbol{x}_l + \boldsymbol{\delta}_l, \boldsymbol{x}_r + \boldsymbol{\delta}_r)$. Although the original model loss function in (5) would alleviate this hazard by computing the errors between the model results and the ground truths, L_b would no longer be a help but a burden.

To counteract this optimization ambiguity, we add a margin $\boldsymbol{m}$ to reinforce the optimization of the distance functions [52], [53]. Take $d(\boldsymbol{x}_l, \boldsymbol{x}_r)$ as an example. $f_l(\boldsymbol{x}_l)$ and $f_r(\boldsymbol{x}_r)$ are in symmetric positions in $d(\boldsymbol{x}_l, \boldsymbol{x}_r)$. This means that adding a positive margin to $f_l(\boldsymbol{x}_l)$ is equivalent to adding a negative margin to $f_r(\boldsymbol{x}_r)$. The margin-based distance functions are shown as follows:

$$\begin{aligned} d(\boldsymbol{x}_l, \boldsymbol{x}_r) &= \|f_l(\boldsymbol{x}_l) - f_r(\boldsymbol{x}_r) + \boldsymbol{m}\|_n \\ d(\boldsymbol{x}_l + \boldsymbol{\delta}_l, \boldsymbol{x}_r + \boldsymbol{\delta}_r) &= \|f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r) + \boldsymbol{m}\|_n. \end{aligned} \tag{9}$$

The same margin $\boldsymbol{m}$ is shared in the two distance metrics because we expect the model to recover the same results after being attacked.

The tasks using the fused features learned from the early module can be modeled as multivariate functions. Denote the multivariate function as $f_m(\boldsymbol{x}_l, \boldsymbol{x}_r)$, and the function with perturbations as $f_m(\boldsymbol{x}_l + \boldsymbol{\delta}_l, \boldsymbol{x}_r + \boldsymbol{\delta}_r)$. We hope the model can get the same result for the attacked images, therefore the loss term to be minimized is defined as follows:

$$L_m = \|f_m(\boldsymbol{x}_l + \boldsymbol{\delta}_l, \boldsymbol{x}_r + \boldsymbol{\delta}_r) - f_m(\boldsymbol{x}_l, \boldsymbol{x}_r)\|_n. \tag{10}$$

Unlike (9), we do not add a margin here since the features learned from the perturbed images should equal the original parts. L_m in (10) is distinct from the original loss function L_o in (5) though they both use the two inputs. Usually, the L_o computes the cross-entropy loss on the final results while L_m computes the distance between the features. Specifically, $f_m(\cdot)$ is the feature tensor in the model rather than the final model outputs.

Both the L_b and L_m can be added to the loss function as the stereo-based regularization terms. Together with the original

loss function L_o in (5), the new optimization objective function is defined as follows:

$$L = L_o + L_b + L_m. \tag{11}$$

Using this L makes our method different from the other general adversarial methods. For the ℓ_n norm in the above formulations, we usually use the ℓ_1 norm for simplicity.

B. Local Smoothness Optimization

Recent work has demonstrated that the robustness of models usually suffers from the nonlinearity of loss surface and gradient obfuscation. To simulate the perturbations to attack the models, violating the linearity of the loss surface is of help [54]–[56]. Therefore, we propose optimizing the problem from the perspective of local smoothness while considering the regularization terms defined above.

L_b defined in (8) is transformed to a formulation shown as follows:

$$\begin{aligned} L_b = \| \| f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r) + \boldsymbol{m} \|_1 \\ - \| f_l(\boldsymbol{x}_l) - f_r(\boldsymbol{x}_r) + \boldsymbol{m} \|_1 \|_1 \end{aligned} \tag{12}$$

where we use the ℓ_1 norm for simplicity. The nested norm parameters are challenging to be solved and $\boldsymbol{m}$ is to be determined before the optimization. Besides, the difference between the two terms in $\|\cdot\|$ is at a high magnitude, while the loss surface usually has a low magnitude. Inspired by recent work which approximates the regularization term by the remainder of its Taylor expansion [54], [55], we propose to relax (12) as (13). Refer to the Appendix for the details of the relaxation process

$$\begin{aligned} L_b &\le \| f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_l(\boldsymbol{x}_l) \|_1 + \| f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r) - f_r(\boldsymbol{x}_r) \|_1 \\ &\le \| \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l) \|_1 + \gamma_l(\boldsymbol{\epsilon}, \boldsymbol{x}_l) \\ &\quad + \| \boldsymbol{\delta}_r \nabla_{\boldsymbol{x}_r} f_r(\boldsymbol{x}_r) \|_1 + \gamma_r(\boldsymbol{\epsilon}, \boldsymbol{x}_r) \end{aligned} \tag{13}$$

where $\boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l)$ is the first-order term in the Taylor expansion of $f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l)$, and $\boldsymbol{\delta}_r \nabla_{\boldsymbol{x}_r} f_r(\boldsymbol{x}_r)$ is of $f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r)$. $\gamma_l(\boldsymbol{\epsilon}, \boldsymbol{x}_l)$ and $\gamma_r(\boldsymbol{\epsilon}, \boldsymbol{x}_r)$ are the maximums of the high-order remainders of the Taylor expansions. They are defined as follows:

$$\begin{aligned} h_l(\boldsymbol{\epsilon}, \boldsymbol{x}_l) &= \| f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_l(\boldsymbol{x}_l) - \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l) \|_1 \\ h_r(\boldsymbol{\epsilon}, \boldsymbol{x}_r) &= \| f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r) - f_r(\boldsymbol{x}_r) - \boldsymbol{\delta}_r \nabla_{\boldsymbol{x}_r} f_r(\boldsymbol{x}_r) \|_1 \\ \gamma_l(\boldsymbol{\epsilon}, \boldsymbol{x}_l) &= \max_{\|\boldsymbol{\delta}_l\|_p \le \boldsymbol{\epsilon}} h_l(\boldsymbol{\epsilon}, \boldsymbol{x}_l), \gamma_r(\boldsymbol{\epsilon}, \boldsymbol{x}_r) = \max_{\|\boldsymbol{\delta}_r\|_p \le \boldsymbol{\epsilon}} h_r(\boldsymbol{\epsilon}, \boldsymbol{x}_r) \end{aligned} \tag{14}$$

where h_l and h_r represent the high-order remainders for the left and right images, respectively.

With (13), we can not only erase $\boldsymbol{m}$ but also relax (12) to its upper bound. Considering the tradeoff between computational workload and model accuracy, the higher order remainders, e.g., the second-order gradient, are not computed. The insights behind (13) are straightforward: the difference between $f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l)$ and $f_l(\boldsymbol{x}_l)$ is constrained by the first-order gradient term and the high-order remainder of the Taylor expansion of $f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l)$. h_l and h_r are suitable measures of how linear the surfaces are within the perturbation range $\boldsymbol{\epsilon}$. This kind of quality is called *local smoothness measure*. By minimizing the smoothness terms, we will enhance the loss surface's smoothness and improve the model's robustness. On the contrary, to attack the model, we can manipulate the input images to maximize the smoothness term to break the smoothness of the loss surface.

As to the multivariate regularizer L_m, it follows a similar relaxation strategy. $f_m(\boldsymbol{x}_l + \boldsymbol{\delta}_l, \boldsymbol{x}_r + \boldsymbol{\delta}_r)$ is approximated by

$$\begin{aligned} f_m(\boldsymbol{x}_l + \boldsymbol{\delta}_l, \boldsymbol{x}_r + \boldsymbol{\delta}_r) \approx f_m(\boldsymbol{x}_l, \boldsymbol{x}_r) + \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_m(\boldsymbol{x}_l, \boldsymbol{x}_r) \\ + \boldsymbol{\delta}_r \nabla_{\boldsymbol{x}_r} f_m(\boldsymbol{x}_l, \boldsymbol{x}_r). \end{aligned} \tag{15}$$

Thus, we can form the following bound:

$$\begin{aligned} L_m \le \left\| \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_m(\boldsymbol{x}_l, \boldsymbol{x}_r) + \boldsymbol{\delta}_r \nabla_{\boldsymbol{x}_r} f_m(\boldsymbol{x}_l, \boldsymbol{x}_r) \right\|_1 \\ + \gamma_m(\boldsymbol{\epsilon}, \boldsymbol{x}_l, \boldsymbol{x}_r) \end{aligned} \tag{16}$$

where $\gamma_m(\boldsymbol{\epsilon}, \boldsymbol{x}_l, \boldsymbol{x}_r)$ is the maximum of the high-order remainder $h_m(\boldsymbol{\epsilon}, \boldsymbol{x}_l, \boldsymbol{x}_r)$. They are defined as follows:

$$\begin{aligned} h_m(\boldsymbol{\epsilon}, \boldsymbol{x}_l, \boldsymbol{x}_r) &= \| f_m(\boldsymbol{x}_l + \boldsymbol{\delta}_l, \boldsymbol{x}_r + \boldsymbol{\delta}_r) - f_m(\boldsymbol{x}_l, \boldsymbol{x}_r) \\ &\quad - \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_m(\boldsymbol{x}_l, \boldsymbol{x}_r) - \boldsymbol{\delta}_r \nabla_{\boldsymbol{x}_r} f_m(\boldsymbol{x}_l, \boldsymbol{x}_r) \|_1 \\ \gamma_m(\boldsymbol{\epsilon}, \boldsymbol{x}_l, \boldsymbol{x}_r) &= \max_{\|\boldsymbol{\delta}_l\|_p \le \boldsymbol{\epsilon}, \|\boldsymbol{\delta}_r\|_p \le \boldsymbol{\epsilon}.} h_m(\boldsymbol{\epsilon}, \boldsymbol{x}_l, \boldsymbol{x}_r). \end{aligned} \tag{17}$$

Combining (14) and (17) together, we define the regularization term for high-order remainder as L_h, as shown as follows:

$$L_h = h_l(\boldsymbol{\epsilon}, \boldsymbol{x}_l) + h_r(\boldsymbol{\epsilon}, \boldsymbol{x}_r) + h_m(\boldsymbol{\epsilon}, \boldsymbol{x}_l, \boldsymbol{x}_r). \tag{18}$$

Similarly, we combine all of the first-order gradient terms together, and then we have the regularization term L_∇ defined as follows:

$$\begin{aligned} L_\nabla = \| \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l) \|_1 + \| \boldsymbol{\delta}_r \nabla_{\boldsymbol{x}_r} f_r(\boldsymbol{x}_r) \|_1 \\ + \| \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_m(\boldsymbol{x}_l, \boldsymbol{x}_r) + \boldsymbol{\delta}_r \nabla_{\boldsymbol{x}_r} f_m(\boldsymbol{x}_l, \boldsymbol{x}_r) \|_1. \end{aligned} \tag{19}$$

By maximizing L_h, we can break the smoothness of the loss surface, so as to generate powerful adversarial images. By minimizing these regularization terms together with the original loss, we can train the model to improve the smoothness and defense the perturbations. Therefore, we rewrite the min–max formulation in (5) and (11) as follows:

$$\begin{aligned} \arg\min_{\boldsymbol{\theta}} \ \mathbb{E}_{(\boldsymbol{x}_l, \boldsymbol{x}_r, \boldsymbol{y}) \in \mathbb{D}} \left[L_o + L_\nabla + \max_{\boldsymbol{\delta}_l, \boldsymbol{\delta}_r \in \Delta} L_h \right] \\ \text{s.t.} \ \ \|\boldsymbol{\delta}_l\|_p \le \boldsymbol{\epsilon}, \|\boldsymbol{\delta}_r\|_p \le \boldsymbol{\epsilon}. \end{aligned} \tag{20}$$

The min–max formulation is computed on batches of input images during the generation of adversarial images and the model training. The method proposed in the section is shorted as `SmoothStereo`. The inner maximization is the smoothness-driven generation of adversarial images, and the outer minimization is the smooth defense.

IV. Smooth Gradient Defense

As mentioned above, `SmoothStereo` consists of two parts: 1) smoothness-driven generation of adversarial images and 2) smooth defense. In this section, we propose to enhance `SmoothStereo` as `SmoothStereoV2` by using the smooth activation functions to introduce smoothness to the model structure and further improve the loss function (20) to defend

the attacks. In other words, the novel SmoothStereoV2 is composed of smoothness-driven image generation and smooth defense with a smooth loss function, as shown in (26).

A. Smooth Gradient Method

During defense training, the model learns the stored perturbations to improve the model's robustness and tune the model parameters. Improving the model *robustness* usually forces the stereo model to deviate from the original distribution of the clean images to the perturbations in the attacks. These perturbations would possibly do not exist in some scenarios. These would result in the degradation of the inference *accuracy* on the images with weak perturbations. Tsipras *et al.* [57] claimed that the model robustness may be odd with the model accuracy, and there exists a tradeoff. Zhang *et al.* [58] identified the tradeoff as a guiding principle in defense training and proposed a novel defense method, composed of an empirical risk minimization term and a regularization term, to push the classification boundary away from the data. Wang *et al.* [59] proposed a dynamic training strategy to gradually increase the convergence quality of the adversarial examples and provide a theoretical guarantee on the defense convergence. Some researchers suggest improving accuracy by enhancing the generalization ability of the model to suit a wide range of various inputs. Yang *et al.* [60] combined dropout with robust training methods to obtain better generalization and proposes to use a novel locally Lipschitz classifier. Balancing the robustness and accuracy is difficult, while these methods only focus on trivial classification tasks and require heavy model training from scratch. The actual application scenarios are intricate, thus making these methods targeting classification tasks not applicable, especially in our complicated stereo-based tasks.

As mentioned in the above section, violating the smoothness of the loss surface generates powerful perturbation samples. Inspired by this, we propose that increasing the smoothness of the model during defense training would help improve the robustness to attacks. In this article, we propose to use a smooth activation function to preserve more details about the gradients to increase the model's smoothness and robustness [40], i.e., smooth gradient learning.

The most widely used activation function in stereo models is ReLU, a nonsmooth function that introduces nonlinearity into the network [3], [4], [7], [8], [10]. Typically, a stereo model is deep and stacks tens of ReLU functions. Using the ReLU function, the activation values that are less than 0 are forced to be 0 to improve the sparsity of the data to help emphasize the crucial data. The ReLU function and its gradient function are shown in (21) and Fig. 4. Any features less than 0 are forced to be zero while their gradient values are also zeros. There is an abrupt change of the gradient at $x = 0$. General model training methods use ReLU to ignore the details deliberately to improve the model's generalizability and avoid overfitting because of the redundancy in the training sets. With these characteristics, ReLU has been used as primary layers in the existing deep learning models, including the stereo models discussed in this article. For example, there are 11 and 34

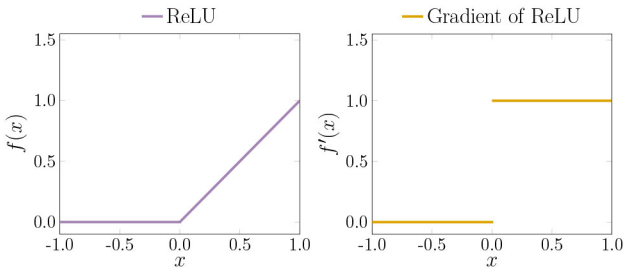


Fig. 4. ReLU function and its gradient function.

ReLU layers in AANet [10] and Stereo R-CNN [8]

$$f(x) = \max(0, x)$$
$$f'(x) = \begin{cases} 0, & x \leq 0 \\ 1, & x > 0. \end{cases} \tag{21}$$

We conjecture that preserving more gradient information help improve the ability to perceive disturbances. However, when $x \leq 0$, the activation values and gradients of ReLU are all equal to 0, and Fig. 4 shows the abrupt changes of the gradient value at $x = 0$. This characteristic is adopted to force to model to forget many details to improve the model's generalizability and avoid overfitting since the deep learning models are usually trained with more than hundreds of epochs on thousands, even millions of images. In comparison, in our problem, a smooth function curve and a smooth gradient curve are preferred to preserve more information since the stereo models ought to be trained to be sensitive enough to the subtle perturbations, in a few training epochs with hundreds of adversarial images and based on the pretrained models. Meanwhile, the models should perform well under various attacks, including weak and strong attacks. The function and gradient values should change continuously, and close to zero should be held and distinguishable. To deal with this problem, we propose using the smooth approximation of the ReLU function, including SoftPlus [34], ELU [35], GELU [36], and Swish [37].

SoftPlus and its gradient function are given as follows:

$$f(\beta, x) = \frac{1}{\beta}\log(1 + \exp(\beta x))$$
$$f'(\beta, x) = \frac{1}{1 + \exp(-\beta x)} \tag{22}$$

where β is a hyperparameter to control the gradient.

ELU takes the smooth form and the gradient function is shown as follows:

$$f(x) = \begin{cases} x, & x \geq 0 \\ \exp(x) - 1, & x < 0. \end{cases}$$
$$f'(x) = \begin{cases} 1, & x \geq 0 \\ \exp(x), & x < 0. \end{cases} \tag{23}$$

GELU weights the inputs nonlinearly by their magnitude, rather than gates inputs by their sign as in ReLUs. GELU takes the form $f(x) = x \cdot \Phi(x)$, where $\Phi(x)$ is the cumulative distribution function of the standard normal distribution. GELU and its gradient function are approximated [36] as (24) to ease

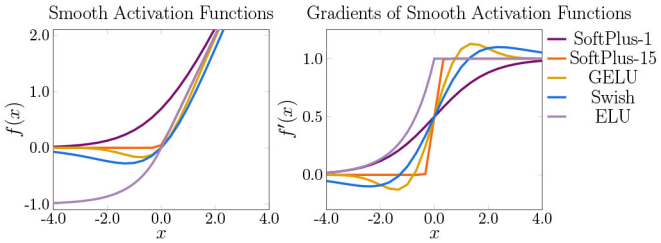


Fig. 5. Smooth activation functions and their gradient functions, with $\beta = 1$ and $\beta = 15$ for SoftPlus.

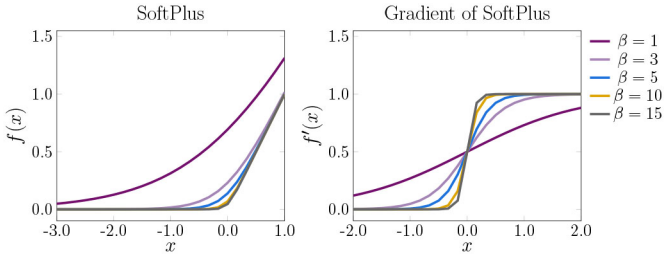


Fig. 6. SoftPlus functions and their gradient functions.

computations

$$f(x) = 0.5x\Big(1 + \tanh\Big[\sqrt{2/\pi}\Big(x + 0.044715x^3\Big)\Big]\Big)$$
$$f'(x) = 0.5\tanh\Big(0.0356774x^3 + 0.797885x\Big) + \Big(0.0535161x^3 + 0.398942x\Big)\operatorname{sech}^2\Big(0.0356774x^3 + 0.797885x\Big) + 0.5. \quad (24)$$

Swish and its gradient function are shown as follows:

$$f(x) = x\sigma(x)$$
$$f'(x) = x\sigma(x) + \sigma(x)(1 - x\sigma(x))$$
$$\text{with } \sigma(x) = (1 + \exp(-x))^{-1}. \quad (25)$$

Compared with ReLU in Fig. 4, abrupt changes do not exist in these smooth activation functions. The values close to zero (which means they are subtle and difficult to perceive) are well reserved. During the defense training with smooth activation functions, the loss value is backpropagated to update the model parameters concerning these small values to make them more sensitive. In contrast, in the nonsmooth ReLU models, their corresponding parameters are ignored and remain unchanged. Using smooth activation functions does not introduce additional model weights, thus keeping the model structures and inference procedures consistent with the ReLU-based models. The computation workloads of the activation function are negligible compared with the computationally heavy layers, such as various types of convolutions and fully connected layers.

Examples of these smooth activation functions and their gradient curves are shown in Fig. 5, with $\beta = 1$ and $\beta = 15$ for SoftPlus. Compared with ReLU in Fig. 4, the functions and gradient functions preserve more information, especially for $x \leq 0$. Though these gradient functions have distinct values for the same x, the differences for different x are highlighted, while in ReLU, there are only two values, 1 and 0.

B. Learning via Smooth Activation Functions

Learning via smooth activation functions helps improve the model robustness to the attacks while considering more to guarantee the performance under various intense attacks. There are two categories of adversarial images, one for defense training and one for testing (i.e., attacks after defense). For SoftPlus with β values, detailed analyses are necessary to handle various defense and testing images.

With the increase of perturbations in the generated adversarial examples, the defense training, as shown in (20), increases the model robustness to stronger attacks during testing. However, the accuracies for test cases with weaker perturbations degrade relatively, or the performance improvements are unsatisfying. Some results are shown in the experiments in Section VI-C. The β value in SoftPlus controls the gradient of SoftPlus and the rate of value changes of the gradient function. The β values are adjusted to facilitate model robustness to strong attacks. If using a smaller β, the curves deviate farther from the x-axis at $x = 0$. As shown in Fig. 6, smaller β values possess more $f(x)$ values if $x < 0$ and have more comprehensive ranges of gradient values to update parameters. For SoftPlus, $\beta = 1$ preserves more information compared with $\beta = 15$. That means more information from the perturbations and the more considerable differences between the clean images and the perturbations are preserved. The differences pull the model weights farther from the original weight distribution during defense training. Other activation functions show more complicated phenomena. For Swish and GELU, more gradient information is preserved compared with other functions. For ELU, the gradient is consistent as 1 if $x \geq 0$. As to the function values shown in Fig. 5, ELU, Swish, and SoftPlus with $\beta = 1$ possess higher diversities compared with GELU. Therefore, in comparison, some critical information is forgotten by GELU during inference.

We need to guarantee that enough information is learned during defense training to enable the model to identify the perturbations. Concerning the perturbation ϵ, if the adversarial training images are generated under a tight constraint (i.e., a small ϵ), using smaller β to preserve more information is preferred since the perturbations are subtle. Suppose the perturbation constraint is loose, i.e., a large ϵ and drastic perturbations. In this case, we prefer larger β values to avoid the strong perturbations to pull the model too far from the original distribution. Empirically, the β value in SoftPlus increases as the perturbations of the adversarial images increase to improve the overall performance and balance the robustness and accuracy.

As mentioned above, Swish and ELU preserve more information, therefore, have higher robustness to weak attacks than GELU. Various SoftPlus functions are better than GELU since more features will be preserved during inference. Compared with the smooth defense training in (20), the smooth gradient method based on smooth activation functions improves the model performance. It provides the opportunities to adjust the balance between robustness and accuracy via β of SoftPlus. Both ReLU and smooth activations do not contain

TABLE I
SUMMARY OF THE PROPOSED METHODS

	Smoothness Generation [+]	Smooth Defense	Smooth Activation Enhancement
`SmoothStereo`	✓	✓	
`SmoothStereoV2`	✓	✓	✓

[+] Smoothness-driven generation of adversarial images

learnable parameters. Although there are some exponential and product operations in smooth activations, the computation overheads are tiny compared to convolutions and FC layers. Some inference latencies are listed in the experimental results to illustrate this.

In summary, the adversarial images are generated by breaking the local smoothness. Then, the generated adversarial images are used to fine-tune the model enhanced by smooth activation functions. For clearness, the min–max optimization formulation in (20) is reformulated as follows:

$$\begin{aligned} \boldsymbol{\delta}_l, \boldsymbol{\delta}_r = & \max_{\boldsymbol{\delta}_l, \boldsymbol{\delta}_r \in \Delta} L_h \\ \arg\min_{\boldsymbol{\theta}} \quad & \mathbb{E}_{(\boldsymbol{x}_l, \boldsymbol{x}_r, \boldsymbol{y}) \in \mathbb{D}} \left[L_a^s = \left(L_o^s + L_\nabla^s + L_h^s \right) \right] \\ \text{s.t.} \quad & \|\boldsymbol{\delta}_l\|_p \leq \boldsymbol{\epsilon}, \|\boldsymbol{\delta}_r\|_p \leq \boldsymbol{\epsilon} \end{aligned} \tag{26}$$

$\max_{\boldsymbol{\delta}_l, \boldsymbol{\delta}_r \in \Delta} L_h$ is discussed in Section III to generate the perturbations based on the original model in which ReLU is used. L_o^s, L_∇^s, and L_h^s are the modified model loss terms by replacing the ReLU in the original model with the smooth activation function, where s denotes *smooth*. In other words, smooth activations are only used to help tune the model parameters during defense training and will not participate in the generation of adversarial images. The enhanced defense method proposed in this section together with the method to generate the adversarial images in Section III is termed `SmoothStereoV2`.

Here, we articulate the strength of the attack. A larger $\boldsymbol{\epsilon}$ will make the images deviate from the clean images farther, resulting in more errors, i.e., stronger perturbations. A smaller $\boldsymbol{\epsilon}$ will lead to weaker perturbations since the deviations are fewer. Besides, PGD is regarded as a stronger attack method than FGSM since PGD will result in more significant performance degradations than FGSM for the same $\boldsymbol{\epsilon}$, as shown in Table IV.

V. Overall Flow

In the previous sections, we discuss the stereo-based local smooth regularizer in detail and introduce the smooth activation function to improve the defense performance. The local smooth regularizer is maximized to generate the adversarial images. The activation function, either SoftPlus, Swish, GELU, or ELU, is adopted in the outer minimization to help the defense training. For clarity, our proposed methods are summarized in Table I. The overall algorithm framework is shown in Algorithm 1. If `SmoothStereoV2` is adopted, smooth activation will be used. β is a hyperparameter for SoftPlus, so we do not treat it as an input parameter of the optimization flow. Without loss of generality, we use s to denote the loss functions with smooth activations as shown in (26), rather than introducing new notations to represent the loss functions for these different activations separately. According to the above discussions, to achieve the optimal defense performance to various attacks, it is preferred to defend through the `SmoothStereoV2` based on adversarial images generated with strong perturbations.

Algorithm 1 Adversarial Training of Stereo-Based Model

Require: Clean image set $\mathcal{D} = \{(\boldsymbol{x}_l^i, \boldsymbol{x}_r^i, \boldsymbol{b}_l^i, \boldsymbol{b}_r^i, y^i)\}_{i=1}^N$, number of samples N, batch size b, iterations of outer minimization T_O, iterations of inner maximization T_I, model parameters $\boldsymbol{\theta}$, learning rate η of model parameters, perturbation range $\boldsymbol{\epsilon}$, step size p of perturbation.

for $t_1 = 1 \rightarrow T_O$ **do**
  Sample a batch $B = \{\boldsymbol{x}_l^i, \boldsymbol{x}_r^i\}_{i=1}^b$ from $\mathcal{D}$;
  Generate random initial perturbation $\Delta = \{\boldsymbol{\delta}_l^i, \boldsymbol{\delta}_r^i\}_{i=1}^b$ for B, under constraint $\boldsymbol{\epsilon}$;
  for $t_2 = 1 \rightarrow T_I$ **do**
    Calculate L_h for the batch B;
    Update Δ via back-propagation to maximize L_h, with perturbation step size p;
  end for
  if Using `SmoothStereo` **then**
    Compute L_o, L_∇, L_h according to B and Δ;
    Compute $L_a = L_o + L_\nabla + L_h$;
    Update $\boldsymbol{\theta}$ via back-propagation to minimize L_a, with learning rate η;
  else if Using `SmoothStereoV2` **then**
    Compute L_o^s, L_∇^s, L_h^s according to B and Δ;
    Compute $L_a^s = L_o^s + L_\nabla^s + L_h^s$;
    Update $\boldsymbol{\theta}$ via back-propagation to minimize L_a^s, with learning rate η;
  end if
end for
return The new model with updated parameters $\boldsymbol{\theta}$.

VI. Experimental Results

A. Experimental Settings

This section evaluates our proposed methods, focusing on the performance of smoothness-driven generation of adversarial images and the smooth gradient defense. The mainstreaming stereo datasets, KITTI 2012 [50] and KITTI 2015 [51], are adopted as the benchmarks. Some stereo-based tasks are tested, including object detection Stereo R-CNN [8] and stereo-matching AANet [10]. Powerful adversarial methods are adopted as the baselines to generate the adversarial images, i.e., FGSM [12], I-FGSM [16], and PGD [17]. The optimal results are bolded. The direct adversarial defense, proposed in [17], is used to defend against adversarial attacks, and the results are compared with our proposed methods. The experimental GPU platform is an Nvidia Titan Xp with CUDA 11.0, and the total global memory is 12196 MB. The CPU platform has 2 Intel Xeon Silver 4114 CPUs. In physical environments, the perturbations should fall in a feasible range. We select some perturbations to test our methods. In experiments, the left and right images share the same perturbation ranges.

TABLE II
RESULTS OF ATTACKING STEREO R-CNN BY FGSM AND PGD

Model	AP_{2d} (%)			AOS (%)			AP_{bv} (%)			AP_{3d} (%)		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
No Attack [8]	99.28	91.09	78.62	98.42	89.43	76.94	54.10	34.44	28.15	68.24	46.84	39.34
FGSM, $\epsilon = 0.7$	88.29	76.45	62.39	87.54	74.11	60.36	40.52	32.94	27.56	15.52	12.19	10.05
FGSM, $\epsilon = 2$	76.82	60.49	49.67	74.73	57.84	47.35	26.21	21.35	16.81	13.64	7.7	6.14
PGD, $\epsilon = 0.7$	69.55	58.94	48.04	66.72	56.04	45.59	22.52	18.88	15.32	7.02	5.53	4.29
PGD, $\epsilon = 2$	53.01	43.11	34.21	51.48	40.23	31.80	9.60	7.61	6.23	3.82	2.22	1.95

Fig. 7. Examples of results on FGSM attacks. The images from left to right are: original detection results (ground truth), adversarial images generated via FGSM with $\epsilon = 2$, defense results via direct adversarial training, and defense results via our SmoothStereo and SmoothStereoV2.

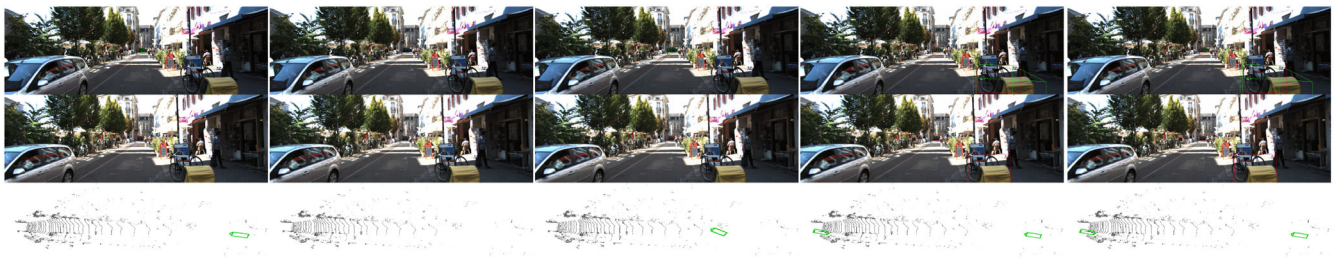


Fig. 8. Example of results on PGD attacks. The images from left to right are: original detection results (ground truth), adversarial images generated via PGD with $\epsilon = 2$, defense results via direct adversarial training, and defense results via our SmoothStereo and SmoothStereoV2.

In the object detection task Stereo R-CNN, we test our proposed methods with two perturbation ranges $\epsilon = 0.7$ and $\epsilon = 2$. In Stereo R-CNN, two sibling branches propose many region proposals for the objects, and then a single branch generates the final object boxes. The two KITTI sets are fused to train and test Stereo R-CNN. In comparison, in the stereo-matching task, AANet matches the features extracted from the two sibling branches to learn the disparity knowledge. Therefore, this task is more sensitive to the changes of features and more vulnerable to attacks than the object detection tasks. We test AANet with more perturbations, $\epsilon = \{0.7, 2, 2.55, 5.1, 10.2\}$. The perturbation step sizes for these ϵ values are $\{0.01275, 0.051, 0.06375, 0.51, 0.765\}$. In the experimental results, SmoothStereoV2 includes Swish, ELU, GELU, and SoftPlus. A set of β values for SoftPlus is tested, i.e., $\{1, 3, 5, 10, 15\}$. The experimental stereo-input images are selected from KITTI sets, following [8] and [10].

B. Smooth Adversarial Methods on Stereo-Object Detection

Some experiments are conducted on object detection Stereo R-CNN [8] to test our methods, compared with FGSM and PGD. The detection performance metrics include AP_{2d} that representing the average detection precision of the 2-D bounding box, AOS that representing the average orientation similarity of the joint 3-D detection, AP_{3d} that representing the average detection precision of the 3-D bounding box, and AP_{bv} that representing the average localization precision of the bird's eye view. The error statistics are computed according to boxes with IoU ≥ 0.7 (IoU). The KITTI object detection set has three categories of inputs: 1) easy; 2) moderate; and 3) hard, which reflect the difficulties of the detection tasks. The perturbation step is 2.

The results of directly using FGSM and PGD to attack Stereo R-CNN are listed in Table II. By applying attacks, the detection qualities degrade significantly.

To improve the model robustness to these attacks, we use direct defense training to tune the model with the FGSM- and PGD-generated images as the defense training set. The defense sets are generated with the same perturbations as the testing sets. The testing results of our proposed methods are listed in Table III. The results show that our methods SmoothStereo and SmoothStereoV2 outperform the baselines significantly in most cases. SmoothStereo wins the baselines, and SmoothStereoV2 with smooth activation functions improves the performance further. The performance improvements are pretty impressive for the instances under PGD attacks, demonstrating the effectiveness of enhancing the model smoothness under strong attacks.

Figs. 7 and 8 show some examples of using FGSM, PGD, and our methods to defend against the attacks. The adversarial

TABLE III
ACCURACY RESULTS OF TESTING STEREO R-CNN AFTER DEFENSE TRAINING

Testing Images	Defense Method	AP_{2d} (%)			AOS (%)			AP_{bv} (%)			AP_{3d} (%)		
		Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
FGSM, $\epsilon = 0.7$	Direct + FGSM	87.58	**81.54**	**71.53**	87.25	**80.11**	62.42	41.95	30.62	28.89	**21.57**	**19.62**	**16.56**
	SmoothStereo	88.38	**82.74**	**73.94**	88.89	**81.87**	63.63	**45.51**	31.01	26.61	**24.50**	**20.88**	**18.26**
	SoftPlus-1	**94.14**	80.09	70.12	**93.08**	78.18	68.48	43.15	**39.12**	**34.00**	16.55	12.49	11.48
	SoftPlus-3	93.63	80.25	70.08	92.64	78.44	68.51	44.16	35.17	31.25	19.09	15.03	12.93
	SoftPlus-5	93.77	80.40	70.19	92.83	78.58	**68.61**	**44.83**	**41.05**	**36.61**	**19.75**	**17.09**	**15.68**
	SoftPlus-10	93.78	80.03	69.86	92.78	78.10	68.14	42.99	38.23	33.24	16.28	15.69	14.15
	SoftPlus-15	93.74	80.09	69.89	92.75	78.16	68.17	**44.30**	**38.97**	33.68	18.32	15.62	13.49
	Swish	93.74	**80.47**	**70.54**	92.76	**78.59**	**68.93**	40.19	35.89	31.91	14.53	13.02	12.03
	ELU	**94.19**	80.30	70.18	**93.04**	78.30	68.40	41.04	38.53	**34.74**	16.20	15.10	13.97
	GELU	**93.92**	80.35	70.19	**92.89**	78.45	**68.52**	40.83	36.21	32.69	16.21	15.36	14.32
FGSM, $\epsilon = 2$	Direct + FGSM	**84.73**	**70.82**	57.90	**84.13**	**69.19**	55.61	**40.15**	**30.57**	**24.42**	**16.21**	**13.03**	**10.54**
	SmoothStereo	**85.95**	**72.64**	**61.22**	**81.65**	**74.83**	**60.00**	**41.43**	**31.63**	**23.79**	**18.25**	**14.76**	**12.53**
	SoftPlus-1	82.27	**69.60**	**60.66**	77.25	**65.71**	57.39	23.01	20.62	18.24	8.21	6.33	6.06
	SoftPlus-3	84.13	68.46	59.75	80.83	64.99	56.92	23.79	21.02	18.35	6.70	6.40	5.63
	SoftPlus-5	81.30	65.23	56.71	78.95	62.33	54.52	22.77	20.73	17.47	4.46	4.29	3.98
	SoftPlus-10	81.98	65.49	57.06	77.93	61.91	54.06	32.53	24.18	21.68	8.91	7.19	6.71
	SoftPlus-15	82.93	66.25	57.47	78.84	62.49	54.48	32.99	24.44	21.78	6.45	6.31	5.34
	Swish	83.37	66.59	59.62	**82.16**	64.20	**57.42**	26.03	21.99	19.22	6.25	6.28	6.07
	ELU	81.37	67.90	**60.54**	79.14	64.86	**57.72**	34.81	25.98	22.54	**10.35**	**8.17**	7.55
	GELU	**84.43**	68.31	59.22	80.25	64.73	56.12	**34.91**	**26.10**	**23.25**	6.50	8.14	**7.66**
PGD, $\epsilon = 0.7$	Direct + PGD	73.37	61.82	56.66	73.04	60.46	50.04	27.47	20.08	18.74	13.77	7.10	9.30
	SmoothStereo	75.67	61.58	59.73	73.43	62.27	52.82	24.88	20.90	16.99	12.44	11.73	9.46
	SoftPlus-1	78.99	70.99	58.59	51.30	41.24	34.42	15.99	14.90	11.55	8.54	5.93	4.55
	SoftPlus-3	**92.24**	**83.59**	**72.60**	**89.07**	**80.54**	**69.64**	**59.75**	**40.67**	**33.53**	**35.77**	**22.65**	**18.43**
	SoftPlus-5	84.51	**79.91**	**67.30**	81.96	**76.69**	**64.33**	45.72	**36.36**	**29.45**	**29.70**	**23.27**	**18.62**
	SoftPlus-10	**87.67**	77.52	65.36	**87.00**	**74.76**	**62.95**	**47.29**	33.45	27.63	23.55	17.22	13.20
	SoftPlus-15	78.73	75.47	63.51	76.49	72.35	60.80	**49.96**	**34.26**	**27.95**	**24.89**	**19.22**	**15.03**
	Swish	**87.48**	**79.85**	66.97	**81.62**	73.43	61.44	39.38	29.58	25.17	24.42	16.87	13.69
	ELU	69.26	60.29	49.39	66.64	57.76	47.16	25.73	21.78	18.03	9.65	7.16	6.25
	GELU	79.26	79.65	**67.33**	77.04	74.07	62.78	34.90	31.75	25.94	19.99	18.13	14.45
PGD, $\epsilon = 2$	Direct + PGD	54.46	49.11	40.44	53.37	46.23	38.07	14.39	12.42	9.43	5.84	4.65	3.29
	SmoothStereo	55.29	49.38	41.92	52.47	47.27	40.60	18.11	10.38	9.32	6.82	4.52	3.94
	SoftPlus-1	56.12	50.01	39.28	33.65	29.44	23.86	16.44	11.54	9.79	5.41	3.20	2.79
	SoftPlus-3	**61.91**	59.03	**50.49**	**61.28**	**57.77**	**49.29**	**38.80**	24.90	20.53	**24.42**	**14.27**	**11.79**
	SoftPlus-5	**70.19**	**68.30**	**55.08**	**68.08**	**65.21**	**52.59**	31.68	**28.80**	**23.24**	12.81	11.23	9.45
	SoftPlus-10	65.95	58.72	47.91	**63.17**	55.59	45.26	**34.31**	**26.11**	20.77	13.00	10.12	8.10
	SoftPlus-15	58.80	58.23	47.67	57.70	55.02	44.79	30.24	23.99	19.06	13.08	10.96	8.73
	Swish	**78.23**	**72.22**	**60.07**	77.28	**70.02**	**57.95**	**44.06**	**32.69**	**25.78**	**23.58**	**17.54**	**13.99**
	ELU	50.72	41.38	34.31	47.95	38.67	31.70	12.96	9.97	8.29	3.26	2.16	1.91
	GELU	59.17	**59.49**	49.18	57.99	55.36	45.71	30.56	25.41	**21.15**	**15.03**	**15.17**	**12.15**

images mislead the model to misclassify cars and incorrectly predict the object orientations. Direct defense training still loses cars while misclassifying the granite steps as a car. In comparison, our methods can correctly predict the locations and directions of the vehicles. Moreover, our regularization and smoothness terms outperform the original Stereo R-CNN detection model in some cases. Fig. 7 shows that our robust models correctly detect cars that were misclassified by the original model. In Fig. 8, our methods detect the vehicle successfully and find the nearest object that hinders the vehicle. Our approach can also overcome the poor performance and instability of Stereo R-CNN.

C. Smooth Adversarial Methods on Stereo Matching

To demonstrate the performance, we test our methods on a more general stereo-matching task, AANet [10]. The perturbation step is equal to 40. The performance metric is the average $L1$ loss between the predicted disparities and the ground truths. The lower loss value reflects better performance.

Without defense training, the results of attacking AANet using the generated adversarial images are listed in Table IV. The $L1$ loss for clean input images is 11.17. Our generated adversarial images seriously mislead the model compared with PGD and FGSM. As the perturbation range ϵ increases, our advantages are more noteworthy. For example, for $\epsilon = 10.20$, the loss value under our SmoothStereo attack is 88.55,

TABLE IV
$L1$ LOSSES OF ATTACKING AANET ON KITTI 2015

Perturbation ϵ	Attack Method	L1 Loss
$\epsilon = 0.70$	FGSM	12.15
	PGD	12.12
	Ours [+]	12.17
$\epsilon = 2.00$	FGSM	12.63
	PGD	14.61
	Ours	15.07
$\epsilon = 2.55$	FGSM	12.93
	PGD	15.85
	Ours	16.43
$\epsilon = 5.10$	FGSM	14.46
	PGD	61.47
	Ours	66.36
$\epsilon = 10.20$	FGSM	17.05
	PGD	80.52
	Ours	88.55

[+] Our Smoothness-driven generation of adversarial images.

9.97% higher than PGD, and 419.35% higher than FGSM. The results also illustrate the strengths of attacks. Larger ϵ values result in stronger attacks, i.e., more significant performance degradations.

We analyze the performance of SmoothStereoV2 with various β values of SoftPlus to illustrate the importance of smooth gradients under attacks. Then, we compare the performance on AANet after applying other defense training methods to highlight our approaches further.

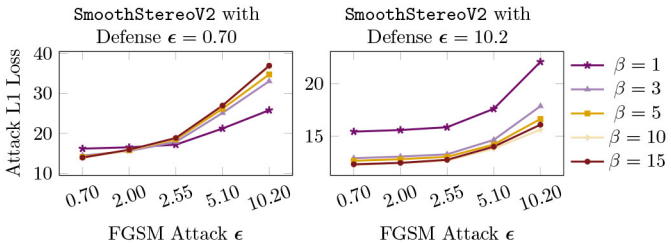


Fig. 9. SmoothStereoV2 defense results with different β values and two defense training sets, under various FGSM attacks.

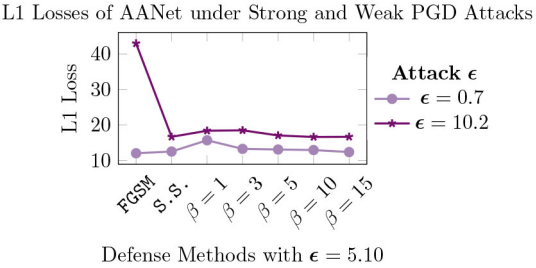


Fig. 10. Defense sets are generated with $\epsilon = 5.10$. FGSM represents FGSM-based defense method. S.S. represents SmoothStereo defense. The β terms represent the SmoothStereoV2 with these β values. $\epsilon = 0.7$ is weak attack and $\epsilon = 10.2$ is strong.

β value of the SoftPlus function plays a vital role during the model inference. By default, researchers set β as 1. We will choose suitable β flexibly to adjust the gradient in SmoothStereoV2. Some results are plotted in Fig. 9. First, increasing the perturbation ϵ of the adversarial defense set would remarkably improve the robustness of the model to defend the attacks. As shown in Fig. 9, the $L1$ loss values are usually higher than 20 with $\epsilon = 0.70$ (i.e., weak attacks in the training set), while most $L1$ loss values are smaller than 15 with $\epsilon = 10.2$ (i.e., strong attacks in the training set). This phenomenon is consistent with empirics that stronger perturbations contribute more to the model's robustness. Besides, if the adversarial image set is generated with $\epsilon = 0.7$, the default $\beta = 1$ has good performance, especially when FGSM attacks are stronger. However, if the adversarial defense set is generated with $\epsilon = 10.2$, the performance of the default $\beta = 1$ is the worst while $\beta = 15$ is optimal. Larger β can always guarantee performance under subtle attacks.

Generally, models perform well under weak attacks, while a satisfying model should still perform well under strong attacks. To illustrate our performance, we plot the results of AANet after defense training under weak PGD attack $\epsilon = 0.7$ and strong PGD attack $\epsilon = 10.2$, as shown in Fig. 10. Under strong and weak FGSM attacks, the experimental results show similar results. It demonstrates that our methods achieve consistent performance under strong and weak attacks while the baselines are inferior under strong attacks.

More results are listed in Table V for KITTI 2012 and KITTI 2015, including different defense methods and defense sets with different perturbations $\epsilon \in \{0.70, 2.00, 2.55, 5.10, 10.20\}$. The result values are the average $L1$ losses of five attacks with attack perturbations $\epsilon \in \{0.70, 2.00, 2.55, 5.10, 10.20\}$. For KITTI 2015, the

TABLE V
$L1$ LOSSES OF ATTACKING AANET

Defense ϵ	Defense Method	KITTI 2015		KITTI 2012	
		FGSM [12]	PGD [17]	FGSM [12]	PGD [17]
$\epsilon = 0.70$	FGSM [12]	27.32	32.61	18.57	**14.80**
	PGD [17]	22.77	29.37	**17.54**	15.00
	I-FGSM [16]	22.48	33.95	18.48	**14.77**
	SmoothStereo	22.45	30.07	19.25	14.92
	SoftPlus-1	**19.39**	**18.10**	19.47	17.16
	SoftPlus-3	21.22	26.10	19.42	15.10
	SoftPlus-5	21.87	29.77	19.41	14.94
	SoftPlus-10	22.43	29.96	19.39	14.97
	SoftPlus-15	22.53	31.40	19.35	15.00
	Swish	**15.35**	**15.90**	18.99	**14.88**
	ELU	**16.93**	**18.51**	**17.68**	15.61
	GELU	27.09	27.38	**17.37**	15.41
$\epsilon = 2.00$	FGSM [12]	21.98	31.70	**29.30**	16.66
	PGD [17]	19.81	21.08	**28.25**	**16.23**
	I-FGSM [16]	19.11	23.43	30.89	16.82
	SmoothStereo	19.42	22.63	34.04	17.14
	SoftPlus-1	19.04	**17.79**	28.53	17.59
	SoftPlus-3	**18.61**	21.88	33.23	16.84
	SoftPlus-5	18.89	22.76	33.95	17.02
	SoftPlus-10	19.15	22.29	35.08	17.09
	SoftPlus-15	19.37	22.56	34.48	17.27
	Swish	**14.68**	**15.03**	31.63	16.40
	ELU	**15.54**	**16.88**	**26.77**	**16.22**
	GELU	24.02	24.24	29.62	**15.91**
$\epsilon = 2.55$	FGSM [12]	17.60	30.41	15.66	18.21
	PGD [17]	17.97	19.24	**15.53**	16.76
	I-FGSM [16]	17.30	19.11	**15.43**	17.34
	SmoothStereo	18.10	20.45	15.56	17.51
	SoftPlus-1	18.65	19.41	17.54	17.65
	SoftPlus-3	17.82	19.81	15.71	16.78
	SoftPlus-5	17.52	19.97	15.59	17.75
	SoftPlus-10	**16.96**	**18.54**	15.59	17.63
	SoftPlus-15	17.91	19.36	15.64	18.11
	Swish	**14.26**	**14.48**	**15.38**	**15.93**
	ELU	**15.10**	**16.33**	15.91	**16.11**
	GELU	24.84	25.02	15.67	**15.81**
$\epsilon = 5.10$	FGSM [12]	14.46	22.56	17.77	25.21
	PGD [17]	15.32	15.08	**16.87**	20.55
	I-FGSM [16]	14.79	14.69	17.56	22.37
	SmoothStereo	15.08	15.12	18.11	22.71
	SoftPlus-1	17.76	16.64	18.84	**18.34**
	SoftPlus-3	15.32	14.94	18.28	20.92
	SoftPlus-5	15.00	14.38	18.23	22.95
	SoftPlus-10	14.79	14.17	18.19	22.79
	SoftPlus-15	**14.21**	**13.82**	18.21	23.62
	Swish	**13.62**	**13.60**	17.87	19.03
	ELU	**13.73**	**13.91**	**17.11**	**17.10**
	GELU	29.15	29.56	**16.77**	**16.77**
$\epsilon = 10.20$	FGSM [12]	14.25	14.75	22.05	30.08
	PGD [17]	14.41	13.70	**20.71**	24.10
	I-FGSM [16]	14.47	14.16	22.55	26.90
	SmoothStereo	14.32	13.94	24.16	27.62
	SoftPlus-1	17.31	16.14	22.80	**19.09**
	SoftPlus-3	14.34	13.73	24.26	24.94
	SoftPlus-5	13.86	13.40	24.30	27.82
	SoftPlus-10	**13.35**	**13.10**	24.65	27.80
	SoftPlus-15	13.51	**13.01**	24.41	28.94
	Swish	**13.23**	13.21	23.33	22.20
	ELU	**13.18**	**13.13**	**20.50**	**18.37**
	GELU	29.81	30.24	**20.72**	**18.13**

results show that with the increasing perturbation ϵ of the defense set, larger β contributes to the lower $L1$ loss. Swish and ELU perform well on both the two KITTI sets, while GELU achieves outstanding results on KITTI 2012. The family of smooth activation functions shows good performance on the strong PGD attacks on these two sets. Some practical stereo-image examples are shown in Fig. 11 with $\beta = 10$. The original model performs poorly on the perturbed images, while after defense with our methods, the display map is of high quality.

In summary, our proposed methods outperform the baselines significantly in improving the model robustness to strong attacks while balancing the accuracies to subtle attacks. As discussed above, traditional models with nonsmooth activation

Fig. 11. Examples of AANet disparity results. The defense and test perturbations are both with $\epsilon = 10.20$. The images from top to down are: original left and right images, perturbed left and right images, disparity map of the perturbed images generated by the original model, and disparity map generated by SmoothStereoV2 with SoftPlus $\beta = 10$. Our disparity map can well reflect the actual physical characteristics while the outputs of the original model are of chaos.

TABLE VI
INFERENCE LATENCIES (MS) ON KITTI 2012

Method	AANet [+]
Original Model	0.30
FGSM Defense	0.29
PGD Defense	0.29
I-FGSM Defense	0.28
SmoothStereo	0.29
SoftPlus-1	0.31
SoftPlus-3	0.31
SoftPlus-5	0.30
SoftPlus-10	0.30
SoftPlus-15	0.29
Swish	0.29
ELU	0.30
GELU	0.31

[+]Each latency is the average value of 6 trials.

functions ignore information during inference and backpropagation to avoid overfitting the redundant data. In our adversarial problems, preserving information to counteract attacks is essential. The model accuracy is also guaranteed since we can handle weak attacks successfully. To achieve the optimal average performance under attacks, we suggest a strong defense set generated by our smoothness-driven generation method with large perturbation $\epsilon = 10.2$, and the smooth defense method SmoothStereoV2 with large β values for SoftPlus (e.g., β = 10 or 15), Swish, or ELU.

D. Analyses on the Inference Costs

The time costs of model inferences are listed in Table VI. We compare the inference latencies of baselines and our methods on KITTI 2012 and AANet. Each latency is the average value of six trials. Results show that using our methods or the baselines will not degrade the real-time inference performance of the stereo model. In deep learning models, the computational workloads arise mainly from the convolutions and fully connected layers. In comparison, the activation layers can be finished quickly, thus impacting the inference speed slightly.

VII. CONCLUSION

To counteract adversarial attacks and improve the robustness of object detection models for autonomous driving systems, novel defense methods that explicitly consider the physical meaning of the stereo-based models are proposed in this article. Our regularization terms can help the model learn the relationships between the left and right images and physical meanings. These terms reflect the local linearity and smoothness of the loss function. Furthermore, a smooth defense method based on the smooth activation function is proposed to improve the model's smoothness during defense training. An optimal defense strategy is summarized based on our smoothness-driven generation of adversarial images and the smooth defense method. It is shown in the results that our methods outperform the baselines significantly. To the best of our knowledge, this is the first work proposing novel methods specifically for the stereo-based problems in autonomous systems.

APPENDIX
RELAXATION OF EQUATION (12)

According to the triangle inequality

$$||a| + |b|| \leq |a \pm b| \leq |a| + |b| \tag{27}$$

which is one of the defining property of the normed vector space [61], (12) can be relaxed to a upper bound

$$\begin{aligned} L_b &= \|\|f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r) + \boldsymbol{m}\|_1 \\ &\quad - \|f_l(\boldsymbol{x}_l) - f_r(\boldsymbol{x}_r) + \boldsymbol{m}\|_1\|_1 \\ &\leq \|f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r) + \boldsymbol{m} \\ &\quad - (f_l(\boldsymbol{x}_l) - f_r(\boldsymbol{x}_r) + \boldsymbol{m})\|_1 \\ &= \|(f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_l(\boldsymbol{x}_l)) - (f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r) - f_r(\boldsymbol{x}_r))\|_1 \\ &\leq \|f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_l(\boldsymbol{x}_l)\|_1 + \|f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r) - f_r(\boldsymbol{x}_r)\|_1. \end{aligned} \tag{28}$$

The left and right images are in the symmetric positions in (28), i.e., $f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r)$ leads to the same deduced results with $f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r) - f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l)$. Furthermore, $f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l)$ can be approximated by its first-order Taylor expansion $f_l(\boldsymbol{x}_l) + \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l)$. Thus, we can have the following bound:

$$\begin{aligned} &\|f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_l(\boldsymbol{x}_l)\|_1 \\ &= \|\boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l) + f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_l(\boldsymbol{x}_l) - \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l)\|_1 \\ &\leq \|\boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l)\|_1 + \|f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_l(\boldsymbol{x}_l) - \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l)\|_1 \\ &\leq \|\boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l)\|_1 + \gamma_l(\boldsymbol{x}_l, \boldsymbol{\epsilon}) \end{aligned} \tag{29}$$

where $\gamma_l(\boldsymbol{x}_l, \boldsymbol{\epsilon})$ is defined as the maximum of the remainder of the first-order Taylor expansion of $f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l)$, i.e.

$$\gamma_l(\boldsymbol{x}_l, \boldsymbol{\epsilon}) = \max_{\|\boldsymbol{\delta}_l\|_p \leq \boldsymbol{\epsilon}} \|f_l(\boldsymbol{x}_l + \boldsymbol{\delta}_l) - f_l(\boldsymbol{x}_l) - \boldsymbol{\delta}_l \nabla_{\boldsymbol{x}_l} f_l(\boldsymbol{x}_l)\|_1. \tag{30}$$

Similarly, the term for the right image is relaxed as follows:

$$\|f_r(\boldsymbol{x}_r + \boldsymbol{\delta}_r) - f_r(\boldsymbol{x}_r)\|_1 \leq \|\boldsymbol{\delta}_r \nabla_{\boldsymbol{x}_r} f_r(\boldsymbol{x}_r)\|_1 + \gamma_r(\boldsymbol{x}_r, \boldsymbol{\epsilon}). \tag{31}$$

Given (29) and (31), L_b is further relaxed to its upper bound as shown as follows:

$$\begin{aligned} L_b &= \|\|f_l(\boldsymbol{x}_l+\boldsymbol{\delta}_l)-f_r(\boldsymbol{x}_r+\boldsymbol{\delta}_r)+\boldsymbol{m}\|_1 \\ &\quad - \|f_l(\boldsymbol{x}_l)-f_r(\boldsymbol{x}_r)+\boldsymbol{m}\|_1\|_1 \\ &\le \|\boldsymbol{\delta}_l\nabla_{\boldsymbol{x}_l}f_l(\boldsymbol{x}_l)\|_1+\gamma_l(\boldsymbol{x}_l,\boldsymbol{\epsilon}) \\ &\quad + \|\boldsymbol{\delta}_r\nabla_{\boldsymbol{x}_r}f_r(\boldsymbol{x}_r)\|_1+\gamma_r(\boldsymbol{x}_r,\boldsymbol{\epsilon}). \end{aligned} \tag{32}$$

REFERENCES

[1] Q. Sun, A. A. Rao, X. Yao, B. Yu, and S. Hu, "Counteracting adversarial attacks in autonomous driving," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des. (ICCAD)*, 2020, pp. 1–7.

[2] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A survey on 3D object detection methods for autonomous driving applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3782–3795, Oct. 2019.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[4] X. Chen, K. Kundu, Y. Zhu, H. Ma, S. Fidler, and R. Urtasun, "3D object proposals using stereo imagery for accurate object class detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 5, pp. 1259–1272, May 2018.

[5] P. Li, T. Qin, and S. Shen, "Stereo vision-based semantic 3D object and ego-motion tracking for autonomous driving," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 646–661.

[6] J.-R. Chang and Y.-S. Chen, "Pyramid stereo matching network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 5410–5418.

[7] S. Duggal, S. Wang, W.-C. Ma, R. Hu, and R. Urtasun, "DeepPruner: Learning efficient stereo matching via differentiable PatchMatch," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 4384–4393.

[8] P. Li, X. Chen, and S. Shen, "Stereo R-CNN based 3D object detection for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 7644–7652.

[9] Y. Chen, S. Liu, X. Shen, and J. Jia, "DSGN: Deep stereo geometry network for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 12536–12545.

[10] H. Xu and J. Zhang, "AANet: Adaptive aggregation network for efficient stereo matching," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 1959–1968.

[11] C. Szegedy *et al.*, "Intriguing properties of neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014.

[12] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015.

[13] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 2574–2582.

[14] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," 2017, *arXiv:1705.07204*.

[15] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, "Adversarial examples for semantic segmentation and object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 1369–1378.

[16] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," 2016, *arXiv:1607.02533*.

[17] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[18] Y. Li, D. Tian, X. Bian, S. Lyu, and M.-C. Chang, "Robust adversarial perturbation on deep proposal-based models," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2018, p. 231.

[19] Y. Li, X. Bian, M.-C. Chang, and S. Lyu, "Exploring the vulnerability of single shot module in object detectors via imperceptible background patches," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2019, p. 218.

[20] S.-T. Chen, C. Cornelius, J. Martin, and D. H. P. Chau, "ShapeShifter: Robust physical adversarial attack on faster R-CNN object detector," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discov. Databases*, 2018, pp. 52–68.

[21] X. Dong *et al.*, "Robust superpixel-guided attentional adversarial attack," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 12895–12904.

[22] K. Eykholt *et al.*, "Robust physical-world attacks on deep learning visual classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 1625–1634.

[23] K. Xu *et al.*, "Adversarial t-shirt! Evading person detectors in a physical world," 2019, *arXiv:1910.11099*.

[24] Z. Wu, S.-N. Lim, L. S. Davis, and T. Goldstein, "Making an invisibility cloak: Real world adversarial attacks on object detectors," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 1–17.

[25] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 284–293.

[26] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[27] T. Bai, J. Luo, J. Zhao, B. Wen, and Q. Wang, "Recent advances in adversarial training for adversarial robustness," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, Aug. 2021, pp. 4312–4321.

[28] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. L. Yuille, "Mitigating adversarial effects through randomization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–16.

[29] N. Das *et al.*, "Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression," 2017, *arXiv:1705.02900*.

[30] J. Lu, T. Issaranon, and D. Forsyth, "SafetyNet: Detecting and rejecting adversarial examples robustly," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 446–454.

[31] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, 2014, pp. 2672–2680.

[32] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 1222–1230.

[33] A. Wong, M. Mundhra, and S. Soatto, "Stereopagnosia: Fooling stereo networks with adversarial perturbations," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, 2021, pp. 2879–2888.

[34] C. Dugas, Y. Bengio, F. Bélisle, C. Nadeau, and R. Garcia, "Incorporating second-order functional knowledge for better option pricing," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, vol. 13, 2001, pp. 472–478.

[35] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," 2015, *arXiv:1511.07289*.

[36] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUs)," 2016, *arXiv:1606.08415*.

[37] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," 2017, *arXiv:1710.05941*.

[38] T. Miyato, A. M. Dai, and I. Goodfellow, "Adversarial training methods for semi-supervised text classification," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–11.

[39] A. Dabouei, S. Soleymani, F. Taherkhani, J. Dawson, and N. M. Nasrabadi, "SmoothFool: An efficient framework for computing smooth adversarial perturbations," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2020, pp. 2665–2674.

[40] C. Xie, M. Tan, B. Gong, A. Yuille, and Q. V. Le, "Smooth adversarial training," 2020, *arXiv:2006.14536*.

[41] E. Rosenfeld, E. Winston, P. Ravikumar, and Z. Kolter, "Certified robustness to label-flipping attacks via randomized smoothing," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2020, pp. 8230–8241.

[42] M. Weber, X. Xu, B. Karlaš, C. Zhang, and B. Li, "RAB: Provable robustness against backdoor attacks," 2020, *arXiv:2003.08904*.

[43] J. Li, Y. Liu, T. Chen, Z. Xiao, Z. Li, and J. Wang, "Adversarial attacks and defenses on cyber–physical systems: A survey," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5103–5115, Jun. 2020.

[44] S. Akiyama and T. Suzuki, "On learnability via gradient method for two-layer ReLU neural networks in teacher-student setting," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2021, pp. 152–162.

[45] H. Asi, J. Duchi, A. Fallah, O. Javidbakht, and K. Talwar, "Private adaptive gradient methods for convex optimization," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2021, pp. 383–392.

[46] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 3723–3732.

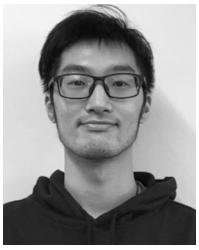
[47] P. Jiang, A. Wu, Y. Han, Y. Shao, M. Qi, and B. Li, "Bidirectional adversarial training for semi-supervised domain adaptation," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2020, pp. 934–940.

[48] C. Sitawarin, S. Chakraborty, and D. Wagner, "SAT: Improving adversarial training via curriculum-based loss smoothing," in *Proc. 14th ACM Workshop Artif. Intell. Security*, 2021, pp. 25–36.

[49] J. Cui, S. Liu, L. Wang, and J. Jia, "Learnable boundary guided adversarial training," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 15721–15730.
[50] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2012, pp. 3354–3361.
[51] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 3061–3070.
[52] Y. Sun *et al.*, "Circle loss: A unified perspective of pair similarity optimization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 6398–6407.
[53] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 2, 2016, pp. 507–516.
[54] C. Qin *et al.*, "Adversarial robustness through local linearization," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, 2019, pp. 13842–13853.
[55] J. Xu, Y. Li, Y. Bai, Y. Jiang, and S.-T. Xia, "Adversarial defense via local flatness regularization," 2019, *arXiv:1910.12165*.
[56] B. Yu, J. Wu, J. Ma, and Z. Zhu, "Tangent-normal adversarial regularization for semi-supervised learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 10676–10684.
[57] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, "Robustness may be at odds with accuracy," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–25.
[58] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 7472–7482.
[59] Y. Wang, X. Ma, J. Bailey, J. Yi, B. Zhou, and Q. Gu, "On the convergence and robustness of adversarial training," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 97, 2019, pp. 6586–6595.
[60] Y.-Y. Yang, C. Rashtchian, H. Zhang, R. Salakhutdinov, and K. Chaudhuri, "A closer look at accuracy vs. robustness," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, 2020, pp. 1–14.
[61] R. Kress, *Numerical Analysis* (Graduate Texts in Mathematics). New York, NY, USA: Springer, 1998. [Online]. Available: https://books.google.com.hk/books?id=e7ZmHRIxum0C

Qi Sun (Graduate Student Member, IEEE) received the B.Eng. degree in computer science from Xidian University, Xi'an, China, in 2018. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

His current research interests include deep neural network hardware acceleration, high-level synthesis, and design space exploration.



Xufeng Yao received the B.Eng. degree in information system and information management from Fudan University, Shanghai, China, in 2016, and the M.Sc. degree in computer science from The Chinese University of Hong Kong, Hong Kong, in 2020, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering.

His research interests include computer vision and machine learning.

Arjun Ashok Rao is currently pursuing the B.Eng. degree in financial technology with The Chinese University of Hong Kong (CUHK), Hong Kong.

He currently works on efficient decentralized learning algorithms with the Systems Engineering Department, CUHK. His research interests broadly include advancing fundamental deep learning results, focusing on adversarial robustness, optimization algorithms for improved generalization, and machine learning for novel science applications.

Bei Yu (Member, IEEE) received the Ph.D. degree from The University of Texas at Austin, Austin, TX, USA, in 2014.

He is currently an Associate Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

Dr. Yu received nine Best Paper Awards from DATE 2022, ICCAD 2021 and 2013, ASPDAC 2021 and 2012, ICTAI 2019, *Integration, the VLSI Journal* in 2018, ISPD 2017, and SPIE Advanced Lithography Conference 2016, and six ICCAD/ISPD contest awards. He has served as the TPC Chair for ACM/IEEE Workshop on Machine Learning for CAD, and in many journal editorial boards and conference committees. He is an Editor of IEEE TCCPS Newsletter.

Shiyan Hu (Senior Member, IEEE) received the Ph.D. degree in computer engineering from Texas A&M University, College Station, TX, USA, in 2008.

He is a Professor and the Chair of Cyber–Physical System Security with the University of Southampton, Southampton, U.K. His research interests include cyber–physical systems and cyber–physical system security, where he has published more than 150 refereed papers.

Prof. Hu is a recipient of the 2017 IEEE Computer Society TCSC Middle Career Researcher Award and the 2014 U.S. National Science Foundation CAREER Award. His publications have received a few distinctions, such as the 2018 IEEE SYSTEMS JOURNAL Best Paper Award, the 2018 IEEE TCSC Most Influential Paper Award, the 2017 Keynote Paper in IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, and the Front Cover Paper in IEEE TRANSACTIONS ON NANOBIOSCIENCE in March 2014. He is the Chair for IEEE Technical Committee on Cyber–Physical Systems. He is the Editor-in-Chief of *IET Cyber–Physical Systems: Theory & Applications*. He is/was an Associate Editor of IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, *ACM Transactions on Design Automation for Electronic Systems*, and *ACM Transactions on Cyber–Physical Systems*. He is/was a Guest Editor of eight IEEE/ACM journals, such as PROCEEDINGS OF THE IEEE and IEEE TRANSACTIONS ON COMPUTERS. He has held chair positions in various IEEE conferences. He is a member of European Academy of Sciences and Arts and a Fellow of IET and British Computer Society. He is an ACM Distinguished Speaker and an IEEE Systems Council Distinguished Lecturer.