sensors

MDPI

Article

Content Swapping: A New Image Synthesis for Construction Sign Detection in Autonomous Vehicles

Hongje Seong, Seunghyun Baik, Youngjo Lee, Suhyeon Lee and Euntai Kim *

School of Electrical and Electronic Engineering, Yonsei University, Seoul 03722, Korea; hjseong@yonsei.ac.kr (H.S.); shbaik104@yonsei.ac.kr (S.B.); lzozo95@yonsei.ac.kr (Y.L.); hyeon93@yonsei.ac.kr (S.L.)

* Correspondence: etkim@yonsei.ac.kr; Tel.: +82-2-2123-7729

Abstract: Construction signs alert drivers to the dangers of abnormally blocked roads. In the case of autonomous vehicles, construction signs should be detected automatically to prevent accidents. One might think that we can accomplish the goal easily using the popular deep-learning-based detectors, but it is not the case. To train the deep learning detectors to detect construction signs, we need a large amount of training images which contain construction signs. However, collecting training images including construction signs is very difficult in the real world because construction events do not occur frequently. To make matters worse, the construction signs might have dozens of different construction signs (i.e., contents). To address this problem, we propose a new method named content swapping. Our content swapping divides a construction sign into two parts: the board and the frame. Content swapping generates numerous synthetic construction signs by combining the board images (i.e., contents) taken from the in-domain images and the frames (i.e., geometric shapes) taken from the out-domain images. The generated synthetic construction signs are then added to the background road images via the cut-and-paste mechanism, increasing the number of training images. Furthermore, three fine-tuning methods regarding the region, size, and color of the construction signs are developed to make the generated training images look more realistic. To validate our approach, we applied our method to real-world images captured in South Korea. Finally, we achieve an average precision (AP_{50}) score of 84.98%, which surpasses that of the off-the-shelf method by 9.15%. Full experimental results are available online as a supplemental video. The images used in the experiments are also released as a new dataset CSS138 for the benefit of the autonomous driving community.

Keywords: construction sign detection; image synthesis; cut-and-paste; perspective transformation

Citation: Seong, H.; Baik, S.; Lee, Y.; Lee, S.; Kim, E. Content Swapping: A New Image Synthesis for Construction Sign Detection in Autonomous Vehicles. *Sensors* **2022**, *22*, 3494. https://doi.org/10.3390/s22093494

Academic Editor: Son Lam Phung

Received: 5 April 2022
Accepted: 1 May 2022
Published: 4 May 2022

1. Introduction

The misdetection of a construction sign may lead to accidents by unexpectedly entering blocked roads. Therefore, the reliable detection of construction signs is quite important in realizing autonomous driving. With the recent progress in object detection based on deep learning [1–6], one might think that we can accomplish the reliable detection of construction signs easily, but it is not true. To train the deep learning detector, we need large-scale training images including construction signs for robust and high-quality results. Unfortunately, construction signs appear infrequently on roads. Thus, collecting large amounts of training data for construction sign detection is required, but it is time-consuming and expensive. To address this problem, we propose a new method for learning to detect construction signs on roads. The main idea of the proposed method is to synthesize training images using a small number of construction sign images. To synthesize training images, we follow the cut-and-paste mechanism [7–9], which cuts an instance from the source image (i.e., construction sign region in an image) and pastes it into a background image (i.e., road image). The cut-and-paste method enables a model to avoid overfitting on a small number of backgrounds in source images, but it cannot generalize limited instances.

A construction sign can be divided into two parts: the board and frame. The content of the sign is contained in a rectangular board, and the board is supported by a frame. The frame can be shared for any sign. Using this characteristic, we effectively generate new construction sign images by swapping the contents in the rectangular board between two different construction sign images, as shown in Figure 1. Our content swapping synthesizes numerous synthetic construction signs by combining the board images (i.e., contents) taken from the in-domain images and the frames (i.e., geometric shapes) taken from the out-domain images. This approach allows us to obtain new $N_I N_O$ images from N_I in-domain construction sign images and N_O out-domain construction sign images. Although in-domain sign images need to be collected using the same camera setting as in the test set, out-domain images can be collected from the Internet. Therefore, we can synthesize a large-scale training dataset with only a small number of in-domain sign images and train a detector on them.

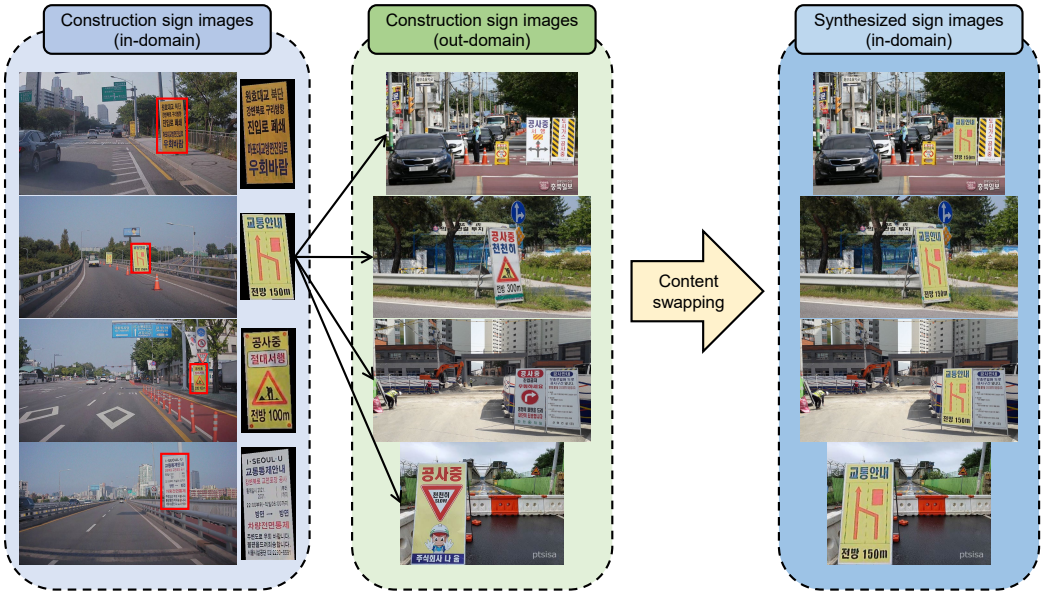

Figure 1. Content swapping. With N_I in-domain and N_O out-domain construction sign images, we synthesize $N_I N_O$ in-domain construction sign images via perspective transformation. The synthesized sign images are used as source images for cut-and-paste.

We also develop three fine-tuning methods to improve the quality of synthetic training images. The three methods deal with the (1) pasted region, (2) instance size, and (3) color difference of the synthesized images, respectively. The first method guides us to paste the synthetic construction sign image on the drivable region. Because the construction sign cannot be placed on the sky, car, or other objects, it should be placed only on the drivable region for realistic purposes. The second method helps us to select the size of the instance based on the location where the sign is to be pasted. If we assume that the construction sign is always pasted on the road and the road is flat, then we can automatically predict the size of the instance in the image. The prediction not only avoids making construction images either too large or too small but also resizes the images to match nearby objects, thereby improving global consistency. Finally, we blend the synthesized construction signs with the training image to reduce the gap between the source and background images. The blending also reduces the domain gap between in-domain and out-domain construction sign images in content swapping. To our best knowledge, no other research has been conducted to detect the construction signs. To validate the effectiveness of the proposed methods, we collect the CSS138 (Construction Signs in Seoul with 138 images) dataset for training and testing construction sign detection. All the images are captured in Seoul, Korea. The CSS138 dataset can be downloaded at https://github.com/Hongje/content-swapping, (accessed on 5 April 2022). In the experiment, we synthesize a large-scale training dataset with only

12 in-domain sign images and achieve a robust and accurate result with an AP_{50} score of 84.98% for CSS138. Our result surpasses off-the-shelf cut-and-paste by 9.15% in the AP_{50} score. Full experimental results are available online: https://youtu.be/us_qso6C5pw, (accessed on 5 April 2022).

The main contributions are summarized as follows:

- This is the first paper which deals with the *construction* sign detection.
- We propose a new image synthesis method, content swapping, to avoid overfitting on limited instances in source images.
- We further present three fine-tunning methods for creating realistic construction images on roads.
- To demonstrate the efficacy of the proposed method, we construct a new dataset, CSS138, for construction sign detection.
- Finally, we achieve an AP_{50} score of 84.98%, creating a gap of 9.15% from the naive cut-and-paste method.

The remainder of this paper is organized as follows. Previous works related to this study are discussed in Section 2. The proposed method for synthesizing construction images is described in Section 3. The experimental results for CSS138 and the analysis are presented in Section 4. Finally, the conclusions are presented in Section 5.

2. Related Work

2.1. Sign Detection

Early methods designed models for detecting signs heuristically. Specifically, Prince et al. [10] design a sign detection algorithm based on a geometrical analysis of the edges and groups of the sign image features. Escalera et al. [11] segment images using color thresholding and then analyze the shape to detect signs. Fang et al. [12] formulate three types of shapes—circular, triangular, and octagonal—to extract the color features of the signs. Shadeed et al. [13] convert the RGB color space to HSV and YUV color spaces and then defined a heuristic algorithm. Loy et al. [14] exploit the symmetric nature and the pattern of the edge of the triangular, square, and octagonal shapes to predict the shape of the sign image. Bahlmann et al. [15] propose a joint color and shape information modeling approach using a set of Haar wavelet features.

Recently, state-of-the-art approaches have used convolutional neural network (CNN)-based supervised models. Shao et al. [16] train CNNs with simplified Gabor filters. Cao et al. [17] use shallow CNNs to classify the traffic signs. Zhang et al. [18] propose a new cascaded R-CNN architecture that includes multiscale attention and imbalanced samples. Liu et al. [19] propose TSingNet, which is based on feature pyramid networks and includes several attention-based modules. Ahmed et al. [20] propose a new DNN-based framework that is robust in detecting traffic signs, even under challenging weather conditions. Zeng et al. [21] propose an improved YOLOv3 architecture for real-time traffic-sign detection. All previous methods considered only traffic or road signs.

The basic difference between general sign detection and construction sign detection is how much training samples are provided. Differently from the large amount of training images in general sign detection, only dozens of training images are given in construction sign detection. Furthermore, collecting the training images for construction signs is much more difficult. The key idea of our method is how to augment the training images and train a detector on them effectively. The purpose to detect and recognize construction signs is to alert the unplanned situations made by road construction. Understandably, commercial autonomous vehicles can handle not only the planned situations but also the unplanned situations. The typical example of the unplanned situation might be the road construction. In this case, the autonomous vehicle may not have to obey the traffic law. For example, our vehicle may have to cross the road following policeman's hand signal, ignoring the traffic sign. The goal of our paper is to handle that kind of unplanned abnormal situation.

Our construction sign detection can also be considered as a special kind of class imbalance problem. We are dealing with only a single class (i.e., construction sign) and

the instances of the class are highly imbalanced with the background instances such as buildings, roads, or pedestrians. The key idea of the paper is to tackle the serious imbalance problem by augmenting the training samples.

2.2. Image Synthesis for Network Training

Several studies [9,22] have synthesized training images with a focus on realism. Furthermore, task-specific image synthesis has also been extensively studied. Dwibedi et al. [7] propose a simple yet effective training image synthesis method that uses cut-and-paste for object detection. Lee et al. [8] propose content transfer, which transfers tail-class content from source to target to address the class imbalance problem in unsupervised domain-adaptive semantic segmentation. Leon et al. [9] synthesize training images by rendering that does not require real-world images. In this paper, we propose methods for synthesizing construction sign images for sign detection. The key idea of our image synthesis is that the contents of the board are taken from in-domain images, whereas the frame is taken from the out-domain (and in-domain) images. Since the frames includes only the geometrical shape of the sign board, they can be collected from any images (out-domain images) without affecting the detection performance. However, since the board images have their own style, the construction sign images taken only from the in-domain images are used to facilitate the synthesis onto the background road images.

3. Method

3.1. Overview

An overview of the proposed method for synthesizing training data is shown in Figure 2. The entire process of synthesizing the training images comprised four main steps. In the first step, we prepared images by collecting construction sign images and road images. As acquiring construction sign images is difficult, we could only prepare a limited number of sign images. Therefore, we collected additional out-domain construction sign images from the Internet. In the second step, the four corners of the content and segmentation mask were labeled in the construction sign images. In the third step, content swapping was performed using these labels. Finally, the training images were generated via the cut-and-paste mechanism using the proposed realistic transformations.

Figure 2. An overview of training data synthesis. The entire process was divided into four steps. First, we collected three types of images: in-domain construction sign images, out-domain construction sign images, and road images. Then, we labeled four corners of the contents and segmented the construction sign images. The labels were then used for content swapping. Finally, a pair of a construction image and a road image was randomly sampled and synthesized via the cut-and-paste mechanism with proposed realistic transformations. The synthesized images are used for training networks.

For a clearer explanation, we provide a pseudo-code of the proposed method in Algorithm 1. Each step in Algorithm 1 matches Figure 2. In the following subsections, we describe the details of each step.

Algorithm 1 Pseudo-code of the proposed method.

Step1: Collecting construction sign and road images

In-domain construction sign image: I_{In}
Out-domain construction sign image: I_{Out}
Road image: I_{Road}

Step2: Labeling bounding box, segment, and four corners of the board

Bounding box labels: $Bbox_{In}$, $Bbox_{Out}$
Segment labels: M_{In}, M_{Out}
Four corners of the board: $\left(\begin{bmatrix} x_I^1 & y_I^1 \end{bmatrix}\begin{bmatrix} x_I^2 & y_I^2 \end{bmatrix}\begin{bmatrix} x_I^3 & y_I^3 \end{bmatrix}\begin{bmatrix} x_I^4 & y_I^4 \end{bmatrix}\right)$, $\left(\begin{bmatrix} x_O^1 & y_O^1 \end{bmatrix}\begin{bmatrix} x_O^2 & y_O^2 \end{bmatrix}\begin{bmatrix} x_O^3 & y_O^3 \end{bmatrix}\begin{bmatrix} x_O^4 & y_O^4 \end{bmatrix}\right)$

Step3: Content swapping

Randomly select content image (source): $S \in In$
Randomly select frame image (target): $T \in \begin{bmatrix} In & Out \end{bmatrix}$
Set content region mask of target image using four corners label: C_T
Compute transformation matrix $\mathcal{T}$: ▷ Section 3.4

$$\mathcal{T} = \begin{bmatrix} x_S^1 & y_S^1 & 1 & 0 & 0 & 0 & -x_S^1 x_T^1 & -y_S^1 x_T^1 \\ 0 & 0 & 0 & x_S^1 & y_S^1 & 1 & -x_S^1 y_T^1 & -y_S^1 y_T^1 \\ x_S^2 & y_S^2 & 1 & 0 & 0 & 0 & -x_S^2 x_T^2 & -y_S^2 x_T^2 \\ 0 & 0 & 0 & x_S^2 & y_S^2 & 1 & -x_S^2 y_T^2 & -y_S^2 y_T^2 \\ x_S^3 & y_S^3 & 1 & 0 & 0 & 0 & -x_S^3 x_T^3 & -y_S^3 x_T^3 \\ 0 & 0 & 0 & x_S^3 & y_S^3 & 1 & -x_S^3 y_T^3 & -y_S^3 y_T^3 \\ x_S^4 & y_S^4 & 1 & 0 & 0 & 0 & -x_S^4 x_T^4 & -y_S^4 x_T^4 \\ 0 & 0 & 0 & x_S^4 & y_S^4 & 1 & -x_S^4 y_T^4 & -y_S^4 y_T^4 \end{bmatrix}^{-1} \begin{bmatrix} x_T^1 \\ y_T^1 \\ x_T^2 \\ y_T^2 \\ x_T^3 \\ y_T^3 \\ x_T^4 \\ y_T^4 \end{bmatrix}$$

Swap content: $I_T^{CS} = \mathcal{T}(I_S) \odot C_T + I_T \odot (1 - C_T)$

Step4: Cut-and-paste with realistic transformations

Randomly select road image (background): $B \in Road$
Compute pasteable region: P_B ▷ Section 3.5.1
Randomly select bottom point of the sign: $\mathbf{p}_1 = \begin{bmatrix} p_1^x & p_1^y \end{bmatrix} \in P_B$
Compute top point of the sign: ▷ Section 3.5.2

$$\mathbf{p}_2 = \begin{bmatrix} p_2^x & p_2^y \end{bmatrix} = \begin{bmatrix} p_1^x & \left(\tan^{-1}\left(\frac{\tan(\alpha \cdot p_1^y + \beta)}{1-h}\right) - \beta\right) \Big/ \alpha \end{bmatrix}$$

Cut sign image I_T^{CS} and paste to road image I_B:
$I_T^{CP} = \text{Cut} - \text{and} - \text{Paste}(I_T^{CS}, M_T, I_B, \mathbf{p}_1, \mathbf{p}_2)$
Transform segment label to $\mathbf{p}_1$ and $\mathbf{p}_2$: $M_T^{CP} = \text{Transform\_mask}(M_T, \mathbf{p}_1, \mathbf{p}_2)$
Transform bounding box label to $\mathbf{p}_1$ and $\mathbf{p}_2$: $Bbox_T^{CP} = \text{Transform\_box}(Bbox_T, \mathbf{p}_1, \mathbf{p}_2)$
Reduce color difference: $I_T^{GP} = \text{GP} - \text{GAN}(I_T^{CS}, M_T^{CS})$ ▷ Section 3.5.3

Output: Synthesized training image: I_T^{GP};
Synthesized training label: $Bbox_T^{CP}$

3.2. Collecting Images

To collect construction sign and road images, we used the FHD390C-USB(D) (Autonomous A2Z, Gyeongsan, South Korea) camera model. This model captures full HD images (1080p) in 30 frames per second. It has a field of view of 60 degrees. We built a data-collecting platform using this camera model, as shown in Figure 3. The camera was installed at a height of 1500mm from the ground and was positioned in front of a platform so that we could collect front-view images of the roads. In total, we collected 138

construction sign images, of which 12 were used for training and the remaining 126 were used for testing. The collected construction sign images were used as in-domain images. In addition, we collected 992 road images that did not contain any construction signs. All the images were captured in Seoul, Korea. Twelve images were used to collect the contents of the construction signs. We also collected an additional 24 construction sign images from the Internet. They were out-domain construction sign images, and they were used to capture the frame of the construction sign boards.

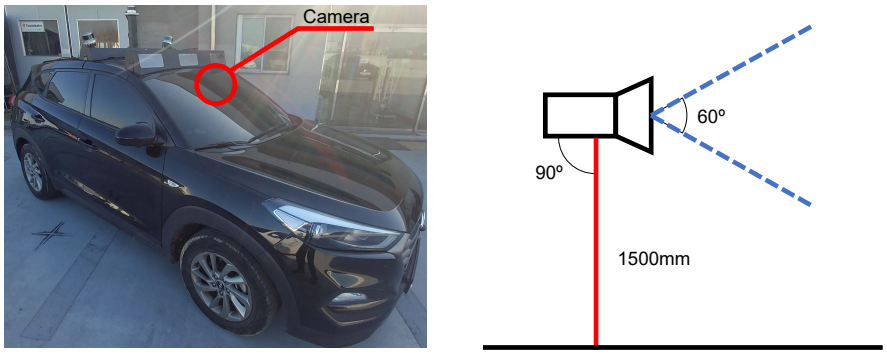

Figure 3. The camera setting in the data-collecting platform.

We collected 12 construction signs using our platform. Thus, we had 12 kinds of construction signs (12 in-domain images) for the board region. We also gathered 24 construction sign images from the Internet, making 36 kinds of construction signs (12 in-domain + 24 out-domain images) for the frame region. The collected 12 in-domain construction signs are shown in Figure 4.

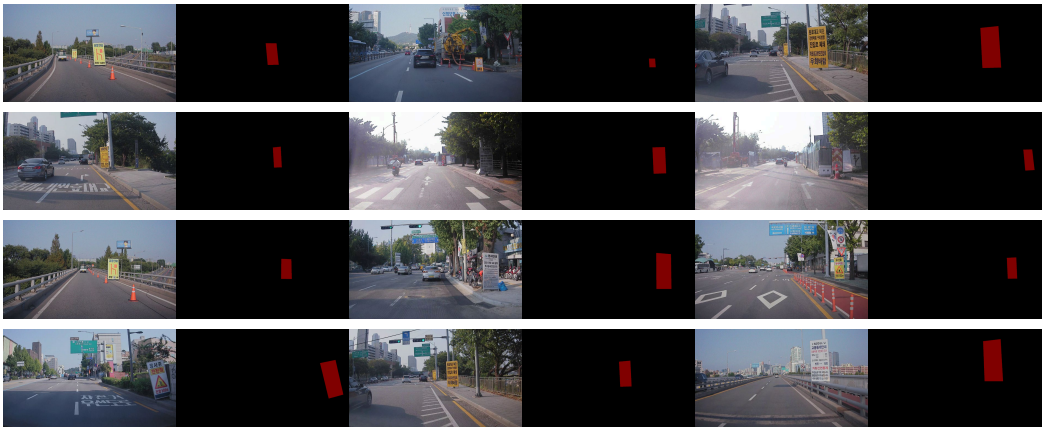

Figure 4. Collected in-domain construction signs.

3.3. Labeling

We annotate three types of labels in the construction sign images. First, we annotate the bounding box for all the collected construction sign images. Bounding box annotations are needed to compute the loss during training and evaluate the detection quality during testing. Second, we annotate the four corners of the board in the training set of construction sign images. Corner annotation is required to calculate the transformation matrix between two construction sign images. Third, we annotate the per-pixel label of the construction sign. Pixel-level annotations are used for both content swapping (detailed in Section 3.4) and cut-and-paste (detailed in Section 3.5).

3.4. Content Swapping

To overcome the lack of the training image, we synthesize training images using a cut-and-paste [7–9] mechanism, as shown in Figure 1. The cut-and-paste effectively helps to prevent the networks from overfitting on the limited backgrounds of the training images. However, the cut-and-paste method cannot augment the content of the training images. This means that only background images can be diversified, and the contents of the construction signs are still limited. We address this problem using content swapping.

The construction sign can be divided into two parts: a rectangular board and frame, as shown in Figure 5. Therefore, we can reuse the frame for other constructions by replacing only the board. To replace the board in the target sign image with the source sign image, we need to formulate the transformation function between the source image and the target image. Thankfully, because the shape of the board is rectangular, replacing the content is possible with four pairs of corner points on the board using perspective transformation, as follows:

$$\begin{bmatrix} wx_T \\ wy_T \\ w \end{bmatrix} = \mathcal{T} \begin{bmatrix} x_S \\ y_S \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & 1 \end{bmatrix} \begin{bmatrix} x_S \\ y_S \\ 1 \end{bmatrix} \tag{1}$$

where $\begin{bmatrix} x_S & y_S \end{bmatrix}^T$ and $\begin{bmatrix} x_T & y_T \end{bmatrix}^T$ are the source and target points of the construction sign images, respectively, and $\mathcal{T} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & 1 \end{bmatrix}$ is the perspective transformation matrix with 8 parameters. Here, the eight parameters in the perspective transformation matrix $\mathcal{T}$ are unknown. Then, we can unfold the equations as:

$$x_T = (p_{11}x_S + p_{12}y_S + p_{13})/w, \tag{2}$$

$$y_T = (p_{21}x_S + p_{22}y_S + p_{23})/w, \tag{3}$$

$$w = p_{31}x_S + p_{32}y_S + 1. \tag{4}$$

By substituting Equation (4) into Equations (2) and (3), we can join a parameter w into x_T and y_T as:

$$x_T = \frac{p_{11}x_S + p_{12}y_S + p_{13}}{p_{31}x_S + p_{32}y_S + 1}, \tag{5}$$

$$y_T = \frac{p_{21}x_S + p_{22}y_S + p_{23}}{p_{31}x_S + p_{32}y_S + 1}. \tag{6}$$

To easily formulate each unknown parameter in $\mathcal{T}$ into a matrix form, we can rearrange Equations (5) and (6) into:

$$x_T = p_{11}x_S + p_{12}y_S + p_{13} - p_{31}x_Sx_T - p_{32}y_Sx_T, \tag{7}$$

$$y_T = p_{21}x_S + p_{22}y_S + p_{23} - p_{31}x_Sy_T - p_{32}y_Sy_T, \tag{8}$$

respectively. Here, there are eight unknown parameters (i.e., $\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{32} \end{bmatrix}$). Therefore, to estimate the eight parameters' values, we need eight different formulas. With Equations (7) and (8), we can make eight different formulas using four known pairs of corre-

sponding points ($\begin{bmatrix} x_S^1 & y_S^1 \end{bmatrix} \cdots \begin{bmatrix} x_S^4 & y_S^4 \end{bmatrix}$ for source points and $\begin{bmatrix} x_T^1 & y_T^1 \end{bmatrix} \cdots \begin{bmatrix} x_T^4 & y_T^4 \end{bmatrix}$ for target points), and then we can write them into a matrix as follows:

$$\begin{bmatrix} x_T^1 \\ y_T^1 \\ x_T^2 \\ y_T^2 \\ x_T^3 \\ y_T^3 \\ x_T^4 \\ y_T^4 \end{bmatrix} = \begin{bmatrix} x_S^1 & y_S^1 & 1 & 0 & 0 & 0 & -x_S^1 x_T^1 & -y_S^1 x_T^1 \\ 0 & 0 & 0 & x_S^1 & y_S^1 & 1 & -x_S^1 y_T^1 & -y_S^1 y_T^1 \\ x_S^2 & y_S^2 & 1 & 0 & 0 & 0 & -x_S^2 x_T^2 & -y_S^2 x_T^2 \\ 0 & 0 & 0 & x_S^2 & y_S^2 & 1 & -x_S^2 y_T^2 & -y_S^2 y_T^2 \\ x_S^3 & y_S^3 & 1 & 0 & 0 & 0 & -x_S^3 x_T^3 & -y_S^3 x_T^3 \\ 0 & 0 & 0 & x_S^3 & y_S^3 & 1 & -x_S^3 y_T^3 & -y_S^3 y_T^3 \\ x_S^4 & y_S^4 & 1 & 0 & 0 & 0 & -x_S^4 x_T^4 & -y_S^4 x_T^4 \\ 0 & 0 & 0 & x_S^4 & y_S^4 & 1 & -x_S^4 y_T^4 & -y_S^4 y_T^4 \end{bmatrix} \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{31} \\ p_{32} \end{bmatrix}. \tag{9}$$

The objective is to estimate eight unknown parameters in $\mathcal{T}$. Therefore, we can finally obtain the transformation matrix $\mathcal{T}$ by computing the inverse of the 8×8 matrix in Equation (9) and performing matrix multiplication as follows:

$$\begin{bmatrix} x_S^1 & y_S^1 & 1 & 0 & 0 & 0 & -x_S^1 x_T^1 & -y_S^1 x_T^1 \\ 0 & 0 & 0 & x_S^1 & y_S^1 & 1 & -x_S^1 y_T^1 & -y_S^1 y_T^1 \\ x_S^2 & y_S^2 & 1 & 0 & 0 & 0 & -x_S^2 x_T^2 & -y_S^2 x_T^2 \\ 0 & 0 & 0 & x_S^2 & y_S^2 & 1 & -x_S^2 y_T^2 & -y_S^2 y_T^2 \\ x_S^3 & y_S^3 & 1 & 0 & 0 & 0 & -x_S^3 x_T^3 & -y_S^3 x_T^3 \\ 0 & 0 & 0 & x_S^3 & y_S^3 & 1 & -x_S^3 y_T^3 & -y_S^3 y_T^3 \\ x_S^4 & y_S^4 & 1 & 0 & 0 & 0 & -x_S^4 x_T^4 & -y_S^4 x_T^4 \\ 0 & 0 & 0 & x_S^4 & y_S^4 & 1 & -x_S^4 y_T^4 & -y_S^4 y_T^4 \end{bmatrix}^{-1} \begin{bmatrix} x_T^1 \\ y_T^1 \\ x_T^2 \\ y_T^2 \\ x_T^3 \\ y_T^3 \\ x_T^4 \\ y_T^4 \end{bmatrix} = \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{31} \\ p_{32} \end{bmatrix}. \tag{10}$$

Using the estimated transformation matrix $\mathcal{T}$, we warped the board from the source image to the target image, which is called content swapping.

With content swapping, we can effectively augment in-domain construction sign images using out-domain construction sign images. Given the N_I in-domain and N_O out-domain construction sign images, we can synthesize in-domain images by content swapping from in-domain sign images to out-domain images, resulting in $N_I N_O$ pairs. Therefore, although we obtained only 12 in-domain construction sign images for training, 288 in-domain images can be obtained using 24 out-domain sign images. Furthermore, we use the frame region in the in-domain sign images for content swapping, which resulted in 432 construction sign images.

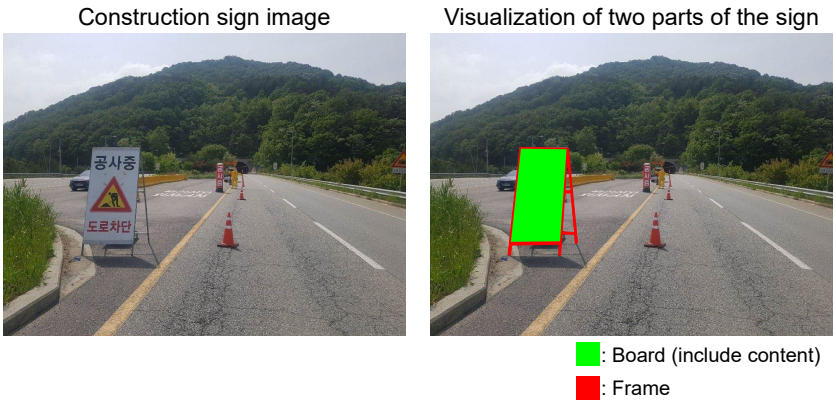

Figure 5. Visualization of the two parts of the construction sign. In the first column, we show a construction sign image. In the second column, we denote the board and frame regions with green and red, respectively.

3.5. Cut-and-Paste with Realistic Transformations

We synthesize training images by cutting a construction sign image and then pasting it onto the background road images. Here, naively cutting and pasting would result in

unrealistic synthetic images, which may lead to performance degradation. We address this problem by proposing three fine-tunning methods. They are developed from three perspectives: pasteable region, instance size, and color difference. Detailed explanations of each fine-tuning methods are provided below.

3.5.1. Pasteable Region

The construction sign cannot fly and is never placed on a car. Therefore, we set the pasteable region as the road. To find road regions in the background image, we used two independent pre-trained networks: semantic segmentation and depth estimation. For the semantic segmentation network, we used DeepLab v3+, trained on Cityscapes https://www.cityscapes-dataset.com, (accessed on 5 April 2022). Because the road class is included in the Cityscapes dataset, the predicted score of the road is used directly. For the depth estimation network, we use the off-the-shelf depth prediction network, MiDaS [23]. The estimated depth is used to filter the noise by thresholding. Thus, the regions that are predicted as roads and with estimated depths lower than the predefined threshold are defined as pasteable regions.

3.5.2. Instance Size

Close objects look large and far objects look small. This property is also preserved in the images. Using this property, we adjust the instance size of the construction sign according to the pasted position. In the image, we first randomly select a pixel within the pasteable region ($\mathbf{p}_1 = \begin{bmatrix} p_1^x & p_1^y \end{bmatrix}$). The selected pixel is the bottom point of the construction sign. In real-world coordinates, we compute the distance between the camera and the sign (d), under the assumption that the road is flat, as follows:

$$d = H_{cam} \tan \theta_1 \tag{11}$$

where H_{cam} denotes the height of the camera from the road, and θ_1 is the angle between the line from the camera to the road and the line from the camera to the bottom of the construction sign line. The angle θ_1 is proportional to p_1^y:

$$\theta_1 = \alpha \cdot p_1^y + \beta \tag{12}$$

where α and β are constants. Given the computed distance d, we can calculate the angle θ_2, which is the angle between the line from the camera to the road and the line from the camera to the top of the construction sign, as follows:

$$\theta_2 = \tan^{-1}\left(\frac{d}{H_{cam} - H_{sign}}\right) \tag{13}$$

where H_{sign} denotes the height of the construction sign, and $H_{sign} < H_{cam}$. For simplification, we assume that all construction signs have the same height H_{sign} and stand perpendicular to the road. Then, in the image coordinates, we compute the top point of the construction sign ($\mathbf{p}_2 = \begin{bmatrix} p_2^x & p_2^y \end{bmatrix}$) using the proportionality between θ_2 and p_2^y as follows:

$$p_2^x = p_1^x, \tag{14}$$

$$
\begin{aligned}
p_2^y &= (\theta_2 - \beta)/\alpha \\
&= \left(\tan^{-1}\left(\frac{d}{H_{cam} - H_{sign}}\right) - \beta\right) \Big/ \alpha \\
&= \left(\tan^{-1}\left(\frac{H_{cam}\tan\theta_1}{H_{cam} - H_{sign}}\right) - \beta\right) \Big/ \alpha \\
&= \left(\tan^{-1}\left(\frac{H_{cam}\tan\left(\alpha \cdot p_1^y + \beta\right)}{H_{cam} - H_{sign}}\right) - \beta\right) \Big/ \alpha.
\end{aligned} \tag{15}
$$

In Equation (15), we divide the denominator and numerator by H_{cam} as:

$$
\begin{aligned}
p_2^y &= \left(\tan^{-1}\left(\frac{\tan\left(\alpha \cdot p_1^y + \beta\right)}{1 - \left(H_{sign}/H_{cam}\right)}\right) - \beta\right) \Big/ \alpha. \\
&= \left(\tan^{-1}\left(\frac{\tan\left(\alpha \cdot p_1^y + \beta\right)}{1 - h}\right) - \beta\right) \Big/ \alpha.
\end{aligned} \tag{16}
$$

where h denotes the ratio of the height of the sign to the camera. Using Equations (14) and (16), the top point of the construction sign can be directly computed from the bottom point. We empirically set the parameters α, β, and h to $\pi/3888$, $\pi/3$, and 0.75, respectively. The overall process for computing the size of the construction signs is summarized in Figure 6.

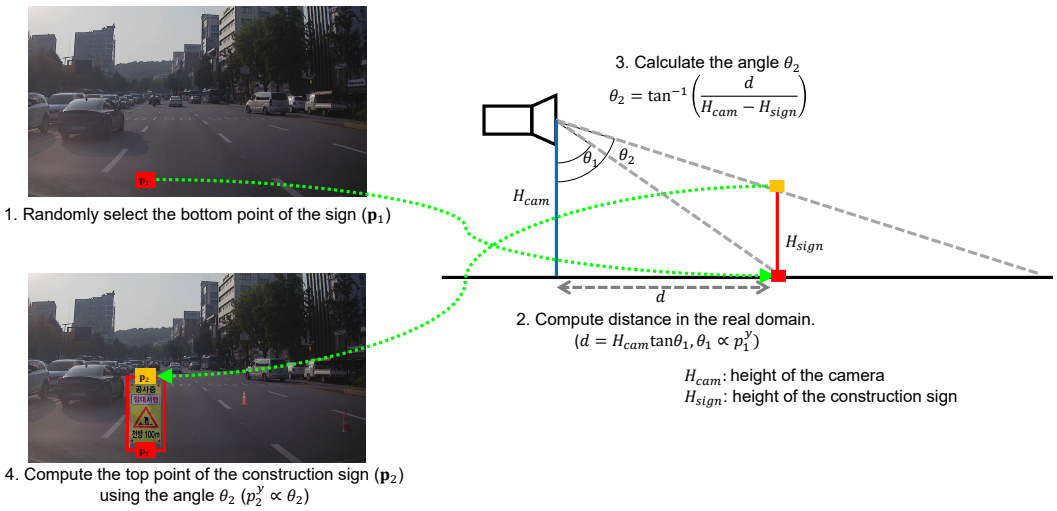

Figure 6. Step-by-step processes for computing the size of the construction sign. (1) We randomly select the bottom position of the construction sign in the image. The position is selected only within the pasteable region. (2) From the randomly selected point in the image, we estimate the angle θ_1 and compute the distance between the camera and the sign in the real domain. (3) We estimate the angle θ_2 by assuming that all construction signs have the same height H_{sign} and stand perpendicular to the road. (4) Using the estimated angle θ_2, we compute the point of the top of the construction sign in the image.

3.5.3. Color Difference

One of the main reasons for the artifacts in the synthesized images, which is made using cut-and-paste, is the color difference between the two images. As shown in Figure 7c, the color difference is caused by differences in illumination, weather, and environment. To match the color difference between the construction sign image and road image, we blend the synthesized image using an off-the-shelf model, GP-GAN [24]. By blending, we can reduce the artifacts of the synthesized image, as shown in Figure 7d.

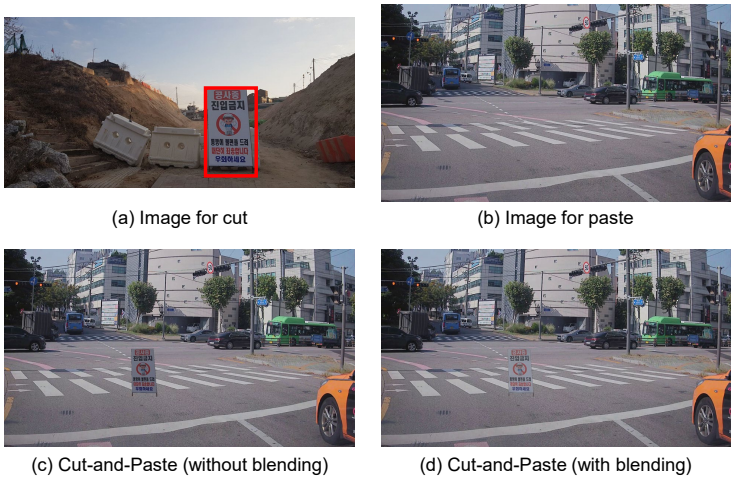

Figure 7. Effect of blending. Using the construction sign image in (**a**) and the road image in (**b**), we synthesize the training image via cut-and-paste. As shown in (**c**), however, the artifact seems prominent because of the color difference between the construction sign and the road. This problem is mitigated by blending, as shown in (**d**).

4. Experiments

4.1. Implementation Details

We conduct some experiments using our collected construction sign detection dataset, CSS138. We use YOLOv3 [1] as a construction sign detector. Basically, we follow the training and inference details in the original YOLOv3 paper [1]. We use Darknet-53 [1] as a backbone network. Darknet-53 consists of 53 convolutional layers and 23 residual connections. Darknet-53 outputs three different sizes of features, which have 1/8, 1/16, and 1/32 resolutions with respect to the input image. To detect construction signs from encoder's feature, a decoder is used. The decoder takes three outputs of Darknet-53, and outputs detection results at three different resolutions, i.e., 1/8, 1/16, and 1/32 resolutions with respect to the input image. Each output predicts five values: four for coordinates of the bounding box and one for objectness. Unlike vanilla YOLOv3, which predicts the class of the object, we do not predict the object class because we have only a single object class, construction sign, in this paper.

Additionally, we apply our method to YOLOv3-tiny to see the effectiveness of our method in other networks. YOLOv3-tiny uses Darknet-19 [25] as a backbone network. Darknet-19 has 19 convolutional layers without residual connections. YOLOv3 has 61.5M parameters, while YOLOv3-tiny has 8.7M parameters. These parameter numbers are comparable with state-of-the-art object detectors: Faster R-CNN [26] has 52.7M parameters, FPN [27] has 60.6M parameters, and RetinaNet [28] has 56.9M parameters. To train YOLOv3, we use an RGB image as an input.

A total of 9920 RGB images are synthesized for training using the CSS138 training set. The training set includes 992 road images and 36 construction sign images. Among 36 sign images, 12 images are in-domain and 24 are out-domain. We randomly crop 640×640 patches for the training. The learning rate is initially set to 1×10^{-2}, and we decrease the learning rate to 1×10^{-3} using the cosine decay schedule. The network is trained in 375,000 iterations with a mini-batch size of 16. The entire training process takes approximately 60 h using a single NVIDIA Titan V GPU. During inference, we obtain multiple detection results at three different resolutions. To select accurate results and dismiss overlapped noisy results, we use non-maximal suppression with an IoU threshold of 0.45.

4.2. Quantitative Results

We validate our approach on the CSS138 validation set. The CSS138 validation set includes 126 images containing at least one construction sign. For quantitative evaluation,

we measure the average precision with Intersection over Union (IoU) thresholds of 0.5, and we denote it as AP_{50}. Following the recent object detection benchmark https://cocodataset.org/#detection-eval, (accessed on 5 April 2022), we additionally measure AP, which is calculated by computing 10 average precision values with IoU thresholds of {0.5, 0.55, ...0.9, 0.95} and then averaging them. To demonstrate the superiority of our approach, we set a baseline that synthesizes 9920 training images by using a naive cut-and-paste method. From the baseline, we add the proposed methods in a step-by-step manner. The experimental results for the CSS138 validation set are listed in Table 1. As shown in Table 1, our method achieves AP and AP_{50} scores of 70.36% and 84.98%, respectively, whereas the baseline achieves scores of 60.53% and 75.84%, respectively. We surpass the baseline by >+9% for both the AP and AP_{50} scores. Table 1 shows the contributions of each step of the proposed method. Each step improves the performance by >+2% for both AP and AP_{50}. This demonstrates that all our approaches are effective in synthesizing training images for construction sign detection.

Table 1. Experimental results on CSS138 validation set.

Method	AP	AP_{50}
A . Baseline (cut-and-paste)	60.53	75.84
B. + Pasteable region	62.74	77.30
C. + Instance size	65.57	82.44
D. + Content swapping	68.51	82.73
E. + Color difference	70.36	84.98

In Table 2, we additionally validate the efficacy of our instance size adjustment method. As described in Section 3.5.2, we resize the instance by projecting it to real-world coordinates. We can compare it with Fixed, which uses the original scale of the construction sign image. We can further compare it with Random, which uses a randomly sampled value for scaling construction sign images and was used in cut-and-paste [7]. As shown in Table 2, we significantly surpass Fixed and Random by 5% and 2%, respectively, in terms of the AP_{50} score. The results demonstrate the superiority of our instance size adjustment method.

Table 2. Analysis experiment on instance size.

Instance Size	AP	AP_{50}
Fixed	62.74	77.30
Random [7]	65.14	80.12
Ours	65.57	82.44

In Table 3, we conduct an ablation study using a different backbone. In the ablation study, we use DarkNet-19 in YOLOv3-tiny. As shown in Table 3, our proposed method improves the detection quality of both YOLOv3-tiny and YOLOv3 networks. This result demonstrates that our proposed method is effective in various networks.

Table 3. Ablation study with various backbone networks.

Method	YOLOv3-tiny (DarkNet-19)		YOLOv3 (DarkNet-53)	
	AP	AP_{50}	AP	AP_{50}
Baseline	53.40	70.67	60.53	75.84
Proposed	54.95	75.57	70.36	84.98

4.3. Grad-CAM Result

In this subsection, we analyze the effectiveness of our proposed method using Grad-CAM. In Figure 8, we visualize the Grad-CAM [29] results of YOLOv3. To extract Grad-CAM, we compute the gradient of the score for objectness at three different resolution

outputs. Then, we average the three activations in each last layer of the decoder. To validate the efficacy of our proposed method, we compare two methods for synthesizing training images. One is to synthesize images simply using naive cut-and-paste method (baseline), and the other one is to synthesize the images using our proposed method. As shown in Figure 8, YOLOv3 trained using a baseline often cannot detect construction signs (second and third rows), while YOLOv3 trained with our proposed method gives accurate activation maps. This result demonstrates that our proposed method helps to learn the discriminative features for construction sign detection.

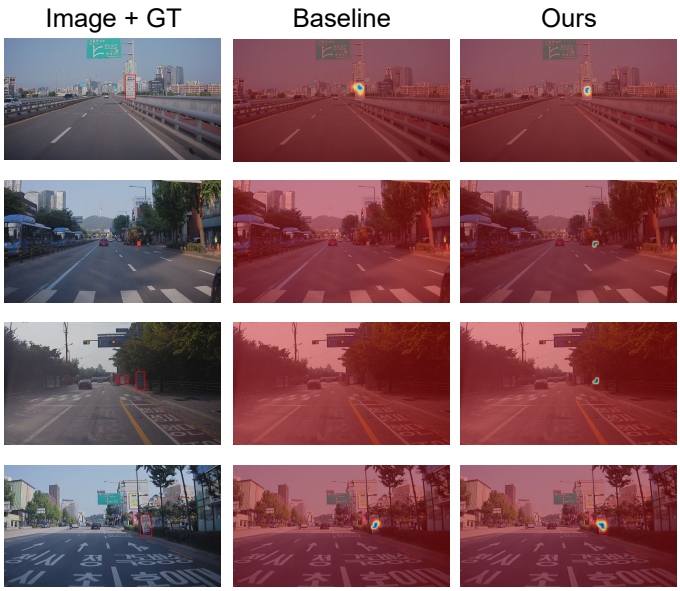

Figure 8. Grad-CAM result. In the first column, input images and corresponding ground truth bonding boxes of the construction sign are shown. In the second and third columns, Grad-CAM results are given. In the Grad-CAM results, high activation values are visualized in blue, while low activation values are visualized in red.

4.4. Effect of Daylight

In this subsection, we analyze the effect of daylight and whether on the performance. We build a hierarchical structure in our CSS138 training set by splitting it into two parts: one is captured under sufficient daylight (i.e., outdoor scene), and the other one is captured under low daylight (i.e., tunnel scene). Among 992 road images, 796 images were taken outdoors and 196 images were taken in tunnel. Examples of outdoor and tunnel scenes are given in Figure 9.

Figure 9. An example of synthesized training set in outdoor and tunnel scenes.

With this split, we train detection networks, and the results for the two different daylight conditions are given in Table 4. As shown in the table, daylight significantly contributed to the performance. Specifically, the performance difference between outdoors and the tunnel is about 30%, and the training set captured under sufficient daylight is more

effective than the one under low daylight in improving detection performance. Therefore, daylight and weather are crucial for construction sign detection.

Table 4. Results on two different daylight conditions.

Split	YOLOv3-tiny (DarkNet-19)		YOLOv3 (DarkNet-53)	
	AP	AP_{50}	AP	AP_{50}
Outdoor	56.34	76.81	69.32	84.88
Tunnel	16.20	32.29	39.91	55.53

4.5. Qualitative Analysis

The synthesized training images are shown in Figure 10. Baseline (*A*) often pastes the construction sign on the sky, which never occurs in real-world scenarios. After considering the pasteable region (*B*), the construction sign is placed on the road, but the scale seems very unfamiliar. Our instance size adjustment method (*C*) could address this problem, but the problem of limited sign images remained. Our content swapping (*D*) effectively augments the construction sign images, preventing overfitting. Finally, the color difference (*E*) between the background road image and foreground construction sign image is adjusted to create a realistic image.

Figure 10. Synthesized training images. For each method, we sampled from the same five background images.

Figure 11 shows the qualitative results of the proposed methods on the CSS138 validation set, as well as the results of the baseline. As shown in the figure, our method finds small instances (first, second, and third rows) and precisely determines the bounding box of the construction sign (fourth row).

In Figure 12, we present some failure cases, and they show some limitation of our method. The first two rows present false negatives, while the last row present false positive. In the first two rows, construction signs are often missed when they are occluded by other objects such as traffic cones. The last row is the example of the false positive. As can be seen, a rectangular shape object is sometimes detected as a construction sign. We expect

that this problem can be solved by various methods, e.g., pre-designing sign shape [10,12], hard example mining [30,31], or learning with strong generalization [32,33].

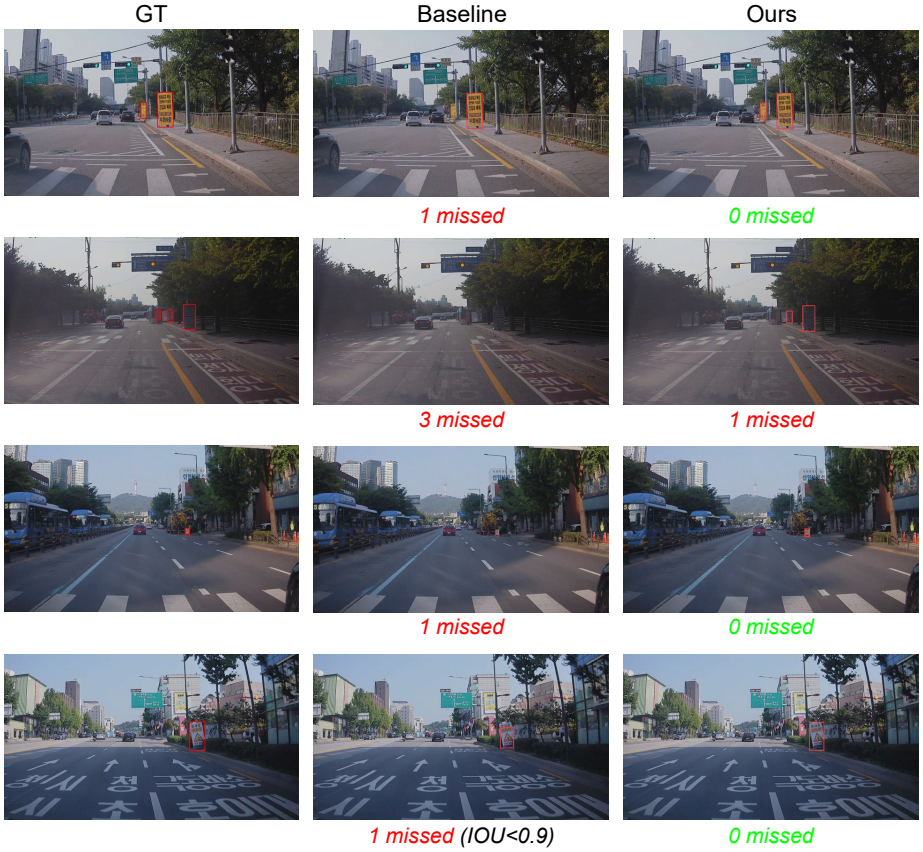

Figure 11. Qualitative results on CSS138 validation set. From left to right, each row shows ground truth (GT), the baseline (naive cut-and-paste), and Ours. For each result on the baseline and ours, we denote the number of missed construction signs.

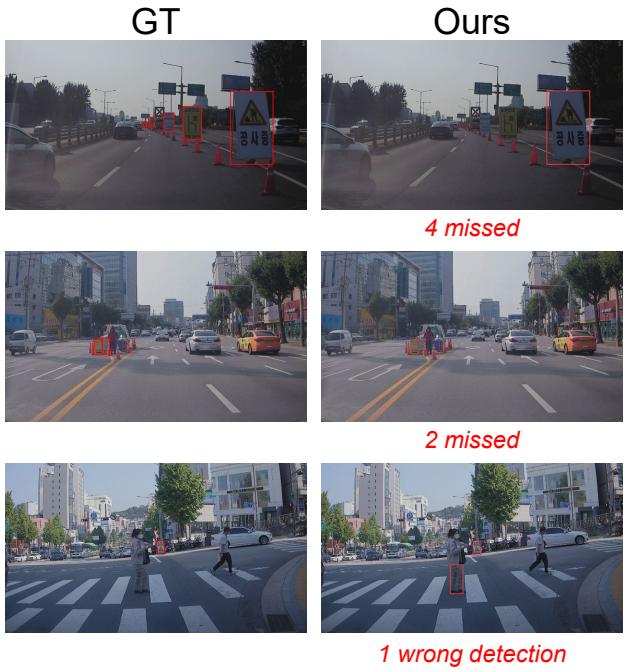

Figure 12. Limitations: if the construction sign is severely occluded, we cannot detect it accurately (first and second rows). A rectangle shape can be detected as a construction sign (third row).

5. Conclusions

In this paper, we have presented a new approach for synthesizing training images for construction sign detection and trained a deep learning detector on them. Since this is the first paper which deals with the construction sign detection, there is not a benchmark set, and we have applied our method to real-world images. Our approach is effective, even when only a few construction sign images are available. Furthermore, our main proposal, content swapping, allows us to use out-domain construction sign data, effectively alleviating the problem of data hunger. To demonstrate the efficacy of our approach, we collected road and construction sign images in person and collected out-domain construction sign images from the Internet. The images used in our experiments are gathered as a dataset CSS138, and we made the dataset available online for the benefit of our community. Even though our method was tested only on the dataset gathered in Seoul, South Korea, we firmly believe that our methods will be applied to other countries and other similar sign-related tasks successfully. Since our content swapping allows us to train networks with a few images, it has the potential to be applied to the few-shot learning field. In this paper, we applied our method only to images, but our proposed method can be extended to videos by applying content swapping and realistic transformations smoothly over time. In addition, our method can be extended to stereo-camera by modeling a construction sign in 3D and projecting it into stereo-view. In addition, a laser scanner sensor can also be considered to measure the distance between the vehicle and the construction sign. The measured distance can improve the quality of the realistic transformations. Furthermore, the future direction of this work would be deciding the action of the autonomous vehicles, after detecting construction signs.

Author Contributions: Conceptualization, H.S., S.L., and E.K.; methodology, H.S., S.L., and E.K.; software, S.B. and Y.L.; validation, H.S., S.B., and Y.L.; formal analysis, H.S., S.L., and E.K.; investigation, H.S., S.L., and E.K.; resources, S.B. and Y.L.; data curation, S.B.; writing—original draft preparation, H.S.; writing—review and editing, S.L. and E.K.; visualization, H.S.; supervision, E.K.; project administration, E.K.; funding acquisition, E.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Technology Innovation Program, (No. 20014124, Development of multipurpose mid-size bus platform technology for automated driving based on predefined route) funded By the Ministry of Trade, Industry and Energy (MOTIE), Korea.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
2. Ahmed, M.; Hashmi, K.A.; Pagani, A.; Liwicki, M.; Stricker, D.; Afzal, M.Z. Survey and Performance Analysis of Deep Learning Based Object Detection in Challenging Environments. *Sensors* **2021**, *21*, 5116. [CrossRef] [PubMed]
3. Zhao, H.; Zhou, Y.; Zhang, L.; Peng, Y.; Hu, X.; Peng, H.; Cai, X. Mixed YOLOv3-LITE: A Lightweight Real-Time Object Detection Method. *Sensors* **2020**, *20*, 1861. [CrossRef] [PubMed]
4. Charouh, Z.; Ezzouhri, A.; Ghogho, M.; Guennoun, Z. A Resource-Efficient CNN-Based Method for Moving Vehicle Detection. *Sensors* **2022**, *22*, 1193. [CrossRef] [PubMed]
5. Miller, D.; Sünderhauf, N.; Milford, M.; Dayoub, F. Uncertainty for Identifying Open-Set Errors in Visual Object Detection. *IEEE Robot. Autom. Lett.* **2021**, *7*, 215–222. [CrossRef]
6. Jiang, L.; Nie, W.; Zhu, J.; Gao, X.; Lei, B. Lightweight object detection network model suitable for indoor mobile robots. *J. Mech. Sci. Technol.* **2022**, *36*, 907–920. [CrossRef]
7. Yun, W.H.; Kim, T.; Lee, J.; Kim, J.; Kim, J. Cut-and-Paste Dataset Generation for Balancing Domain Gaps in Object Instance Detection. *IEEE Access* **2021**, *9*, 14319–14329. [CrossRef]

8. Lee, S.; Hyun, J.; Seong, H.; Kim, E. Unsupervised Domain Adaptation for Semantic Segmentation by Content Transfer. In Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence, virtual, 2–9 February 2021.
9. Eversberg, L.; Lambrecht, J. Generating Images with Physics-Based Rendering for an Industrial Object Detection Task: Realism versus Domain Randomization. *Sensors* **2021**, *21*, 7901. [CrossRef] [PubMed]
10. Prince, S.; Bergevin, R. Road sign detection and recognition using perceptual grouping. In Proceedings of the International Symposium on Automotive Technology & Automation, Florence, Italy, 16–19June 1997.
11. De La Escalera, A.; Moreno, L.E.; Salichs, M.A.; Armingol, J.M. Road Traffic Sign Detection and Classification. *IEEE Trans. Ind. Electron.* **1997**, *44*, 848–859. [CrossRef]
12. Fang, C.Y.; Chen, S.W.; Fuh, C.S. Road-Sign Detection and Tracking. *IEEE Trans. Veh. Technol.* **2003**, *52*, 1329–1341. [CrossRef]
13. Shadeed, W.; Abu-Al-Nadi, D.I.; Mismar, M.J. Road traffic sign detection in color images. In Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003, Sharjah, United Arab Emirates, 14–17 December 2003; Voume 2, pp. 890–893.
14. Loy, G.; Barnes, N. Fast shape-based road sign detection for a driver assistance system. In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566), Sendai, Japan, 28 September–2 October 2004; Volume 1, pp. 70–75.
15. Bahlmann, C.; Zhu, Y.; Ramesh, V.; Pellkofer, M.; Koehler, T. A system for traffic sign detection, tracking, and recognition using color, shape, and motion information. In Proceedings of the EE Proceedings. Intelligent Vehicles Symposium, Las Vegas, NV, USA, 6–8 June 2005; pp. 255–260.
16. Shao, F.; Wang, X.; Meng, F.; Rui, T.; Wang, D.; Tang, J. Real-time traffic sign detection and recognition method based on simplified Gabor wavelets and CNNs. *Sensors* **2018**, *18*, 3192. [CrossRef] [PubMed]
17. Cao, J.; Song, C.; Peng, S.; Xiao, F.; Song, S. Improved traffic sign detection and recognition algorithm for intelligent vehicles. *Sensors* **2019**, *19*, 4021. [CrossRef] [PubMed]
18. Zhang, J.; Xie, Z.; Sun, J.; Zou, X.; Wang, J. A cascaded R-CNN with multiscale attention and imbalanced samples for traffic sign detection. *IEEE Access* **2020**, *8*, 29742–29754. [CrossRef]
19. Liu, Y.; Peng, J.; Xue, J.H.; Chen, Y.; Fu, Z.H. TSingNet: Scale-aware and context-rich feature learning for traffic sign detection and recognition in the wild. *Neurocomputing* **2021**, *447*, 10–22. [CrossRef]
20. Ahmed, S.; Kamal, U.; Hasan, M.K. DFR-TSD: A deep learning based framework for robust traffic sign detection under challenging weather conditions. *IEEE Trans. Intell. Transp. Syst.* **2021**, 1–13. [CrossRef]
21. Zeng, H. Real-Time Traffic Sign Detection Based on Improved YOLO V3. In Proceedings of the 11th International Conference on Computer Engineering and Networks, Beijing, China, 9–11 December 2022; pp. 167–172.
22. Frolov, V.; Faizov, B.; Shakhuro, V.; Sanzharov, V.; Konushin, A.; Galaktionov, V.; Voloboy, A. Image Synthesis Pipeline for CNN-Based Sensing Systems. *Sensors* **2022**, *22*, 2080. [CrossRef] [PubMed]
23. Ranftl, R.; Lasinger, K.; Hafner, D.; Schindler, K.; Koltun, V. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *44*, 1623–1637. [CrossRef] [PubMed]
24. Wu, H.; Zheng, S.; Zhang, J.; Huang, K. Gp-gan: Towards realistic high-resolution image blending. In Proceedings of the ACM Multimedia 2019 Conference, Nice, France, 21–25 October 2019; pp. 2487–2495.
25. Redmon, J.; Farhadi, A. YOLO9000: better, faster, stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 22–25 July 2017; pp. 7263–7271.
26. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *NIPS* **2015**, *28*, 91–99. [CrossRef] [PubMed]
27. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 22–25 July 2017; pp. 2117–2125.
28. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
29. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *Int. J. Comput. Vis.* **2020**, *128*, 336–359. [CrossRef]
30. Shrivastava, A.; Gupta, A.; Girshick, R. Training region-based object detectors with online hard example mining. In Proceedings of the 6 IEEE Conference on Computer Vision and Pattern Recognition, Negas, NV, USA, 27–30 June 2016; pp. 761–769.
31. Lee, S.; Seong, H.; Lee, S.; Kim, E. Correlation Verification for Image Retrieval. In Proceedings of the 2022 Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022.
32. Zhou, K.; Yang, Y.; Qiao, Y.; Xiang, T. Domain Generalization with MixStyle. In Proceedings of the 9th International Conference on Learning Representations, Virtual, 3–7 May 2021.
33. Lee, S.; Seong, H.; Lee, S.; Kim, E. WildNet: Learning Domain Generalized Semantic Segmentation from the Wild. In Proceedings of the 2022 Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022.