Received: 18 March 2023 | Revised: 11 July 2023 | Accepted: 9 August 2023

DOI: 10.4218/etrij.2023-0100

SPECIAL ISSUE

ETRI Journal WILEY

Assembling three one-camera images for three-camera intersection classification

Marcella Astrid[1,2,3] | **Seung-Ik Lee**[1,2]

[1]Department of Artificial Intelligence, University of Science and Technology, Daejeon, Republic of Korea

[2]Field Robotics Research Section, Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea

[3]Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, Luxembourg, Luxembourg

Correspondence
Seung-Ik Lee, Field Robotics Research Section, Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea.
Email: the_silee@etri.re.kr

Funding information
This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (RS-2023-00215760, Guide Dog: Development of Navigation AI Technology of a Guidance Robot for the Visually Impaired Person, 70%) and (No. 2019-0-01309, Development of AI Technology for Guidance of a Mobile Robot to its Goal with Uncertain Maps in Indoor/Outdoor Environments, 30%).

Abstract
Determining whether an autonomous self-driving agent is in the middle of an intersection can be extremely difficult when relying on visual input taken from a single camera. In such a problem setting, a wider range of views is essential, which drives us to use three cameras positioned in the front, left, and right of an agent for better intersection recognition. However, collecting adequate training data with three cameras poses several practical difficulties; hence, we propose using data collected from *one camera* to train a *three-camera* model, which would enable us to more easily compile a variety of training data to endow our model with improved generalizability. In this work, we provide three separate fusion methods (feature, early, and late) of combining the information from three cameras. Extensive pedestrian-view intersection classification experiments show that our feature fusion model provides an area under the curve and F1-score of 82.00 and 46.48, respectively, which considerably outperforms contemporary three- and one-camera models.

KEYWORDS
augmentation, computer vision, deep learning, intersection classification, transfer learning

1 | INTRODUCTION

In autonomous agent navigation, it is vital that the agent be capable of identifying its relative location at all times. Notably, when the agent enters an intersection, it must take appropriate actions, such as turning or continuing forward. Several studies have provided methods of classifying intersections using still images or videos [1–6]; most of these studies use one-camera data. However, the one-camera approach fails to detect the intersection when the agent is inside the intersection as it can no longer detect the notable features, as illustrated in Figure 1. Furthermore, the single camera is inherently incapable of measuring distances; hence, the autonomous agent will have difficulty timing its motions and logistics, as illustrated in Figure 1A. In this work, we overcome these problems by using three cameras placed in the front, left, and right of the agent. As illustrated in Figures 1C and 2, the three simultaneous views give the agent the required decision information, even when traversing the middle of the intersection.

However, this adds new burdens related to fusing the imagery from three sources. First, collecting sufficient three-camera training data would be difficult in terms of equipment, transportation, and power; moreover, it would require extensive human labor in the field. Second, synchronization among the three cameras is required. To cope with these practical difficulties, we propose collecting single-camera data to train the three-camera model.

In addition to simplifying data collection, our proposed augmentation technique uses the power of permutations via the combination of three randomly selected images, which is far less expensive than the alternatives. Additionally, we can easily achieve class balance at no cost just by constructing *intersection* and *non-intersection* data equally. Moreover, the direct collection of three-camera data would produce considerably more *non-intersection* data than *intersection* data, which would create strongly imbalanced data classifications that can potentially impair the model's performance. Lastly, the dataset can be largely constructed and/or expanded using the vast amounts of one-camera data available on the Internet.

The contributions of our work are as follows. (1) We introduce a three-camera scheme for intersection classification to determine whether an agent is in the middle of intersection. (2) We propose an augmentation technique for utilizing one-camera data to train a three-camera model. (3) We propose three methods of combining the information from the three cameras. (4) We perform extensive experiments using various setups to evaluate the importance of our model and justify our design choice.

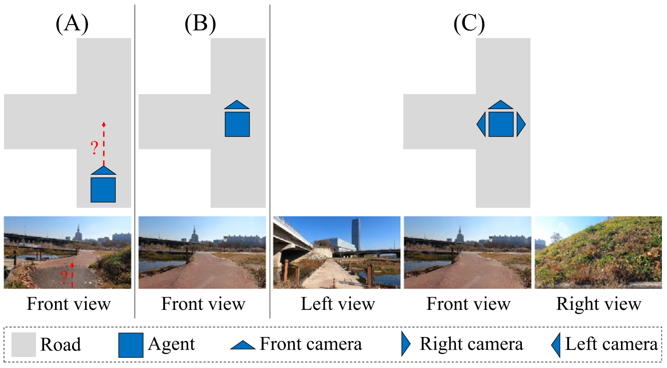

FIGURE 1 Single- versus three-camera view. (A) The agent has difficulty determining the distance to the middle of the intersection. (B) Upon entering the intersection, the front view no longer shows the intersection's features. (C) Our three-camera (front, left, and right) agent view that tracks intersection features from multiple perspectives.

2 | RELATED WORKS

2.1 | Intersection classification

Several intersection classification models have been tested using a variety of external input, such as images [1–7], videos [8], global positioning system (GPS) data [9], light detection and ranging [10, 11], or combinations thereof [2, 12–15]. In this work, we use images as input owing to the simplicity and low cost (e.g., hardware, labor, and computation). Although GPS sensors may be helpful, they routinely lack precision and fail to operate in indoor environments [7].

Conversely, relying solely on one-camera image data invokes the depth perception and loss-of-periphery problems described above (see Section 1). Seff and

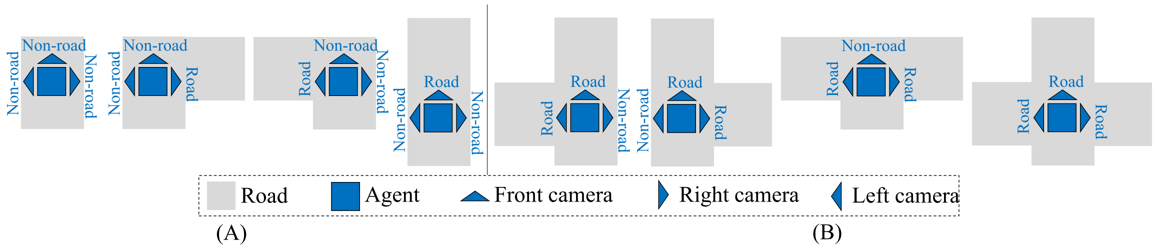

FIGURE 2 Intuition of our three-camera intersection classification. Based on the imagery taken from the three cameras, (A) if fewer than two *road* structures are seen, the scene is classified as *non-intersection*. (B) If two or more road structures are seen from three cameras, the scene is classified as *intersection*.

Xiao [16] built a model that predicts the distance to the middle of intersections using distance labels constructed from Google Street View images. However, those data are taken from GPS receivers [17]. Our proposed method uses three one-camera images to form three-camera training data and does not require GPS information.

2.2 | Augmentation

Data augmentation techniques are widely used to increase the variety and quantity of training data with the goal of improving the generalizability of image classification models [18, 19]. Typically, the augmented data share the same categories as the original data. However, in our method, we create new categories (*intersection* and *non-intersection*) from one-camera *road* and *non-road* classifications.

Several augmentation methods of fusing multiple data into a single training datum have been proposed [20–22]. Specifically, these techniques create a new single image via the combination of multiple images. Our method differs from contemporary methods in that we create a triplet image. These techniques are further explored in Section 4.3.9.

2.3 | Transfer learning

Transfer learning is used to train a model from a source domain to a new target domain, where the source and target data are assumed to come from dissimilar distributions and tasks [23, 24]. This would imply different input spaces and dimensionalities, which is not necessary for our task. Our proposed augmentation technique assumes that the augmented one-camera data have the same distribution as the three-camera data. As such, we leverage an ImageNet-pretrained model to transfer knowledge from 1000 generic image into the three-camera training set.

3 | METHODOLOGY

Our augmentation technique is described in Section 3.1, and the three separate combination methods are explained in Section 3.2.

3.1 | Augmenting one-camera data into three-camera data

Given one-camera data inputs (Figure 3A), labeled as either *road* or *non-road*, we mix and match the types to generate our triplets, each consisting of left X_L, center X_C, and right X_R images (Figure 3B). Algorithm 1 explains the process. We adjust the probability of obtaining *intersection* data using hyperparameter p. By default, we set $p = 0.5$ for a 50% chance of synthesizing intersection or non-intersection triplets to ensure balanced sampling. In our setup, we must also balance the dataset depending on the distribution of *road* images. Hence, a *non-intersection* triplet includes two or three road inputs (Figure 2A), and a non-intersection sample includes zero or one (Figure 2B).

Another sampling method uniformly samples the eight cases illustrated in Figure 2. A comparison of the aforementioned method to Algorithm 1 is provided in Section 4.3.5. This essential level of sampling flexibility is not possible when directly using real-world three-camera data.

In terms of augmentation, the number of intersection classification data that can be obtained from k one-camera items is k^3. This equates to one-billion data when $k = 1000$.

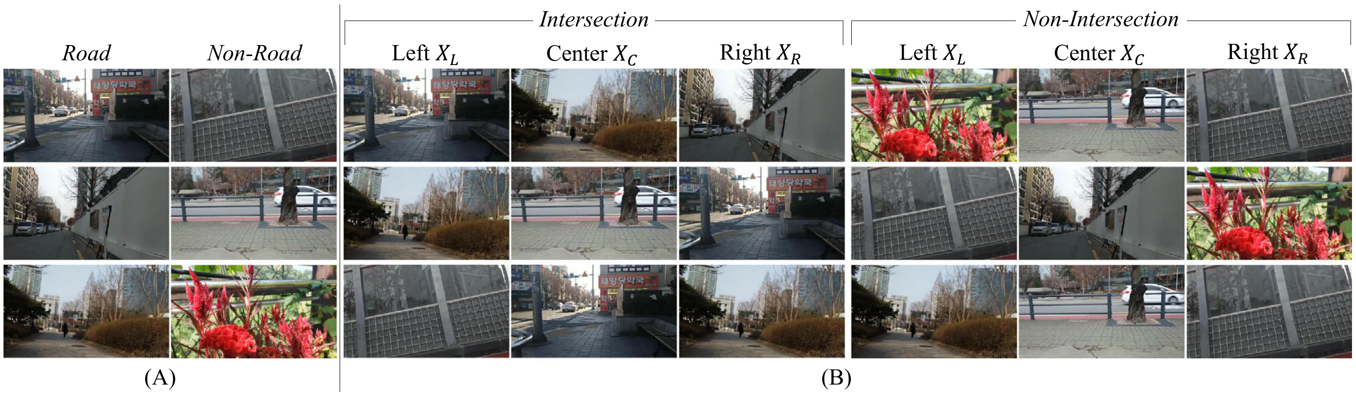

FIGURE 3 Examples of (A) one-camera data and (B) three-camera data generated via one-camera data fusion. Even more three-camera data can be generated via data augmentation techniques.

Algorithm 1 Augmenting one-camera data to create three-camera data. We first randomly decide the final label, y (Line 1). For each *intersection*/*non-intersection* label, we then randomly decide the number of road images in the triplet (Lines 3 and 24), followed by randomly deciding their positions based on the number of roads being two or one (for an intersection triplet) or zero or one (for a non-intersection triplet; see Lines 5 and 27). L, C, and R denote left, center, and right, respectively. Given the selected information, triplet images X_L, X_C, and X_R are sampled appropriately.

```
Require: One-camera data with road and non-road classes.
Ensure: $p$: probability of producing intersection class.
 1: label $y \leftarrow$ intersection with a random probability, $p$.
    Else, $y \leftarrow$ non-intersection.
 2: if $y =$ intersection then
 3:     number_of_road ← 2 or 3 (random)
 4:     if number_of_road = 2 then
 5:         non_road_location ← $L$ or $C$ or $R$ (random)
 6:         if non_road_location = $L$ then
 7:             $X_L \leftarrow$ a randomly selected non-road datum
 8:             $X_C \leftarrow$ a randomly selected road datum
 9:             $X_R \leftarrow$ a randomly selected road datum
10:         else if non_road_location = $C$ then
11:             $X_L \leftarrow$ a randomly selected road datum
12:             $X_C \leftarrow$ a randomly selected non-road datum
13:             $X_R \leftarrow$ a randomly selected road datum
14:         else                              ▷ non_road_location = $R$
15:             $X_L \leftarrow$ a randomly selected road datum
16:             $X_C \leftarrow$ a randomly selected road datum
17:             $X_R \leftarrow$ a randomly selected non-road datum
18:         end if
19:     else                                  ▷ number_of_road = 3
20:         $X_L \leftarrow$ a randomly selected road datum
21:         $X_C \leftarrow$ a randomly selected road datum
22:         $X_R \leftarrow$ a randomly selected road datum
23:     end if
24: else                                      ▷ $y =$ non-intersection
25:     number_of_road ← 0 or 1 (random)
26:     if number_of_road = 1 then
27:         road_location ← $L$ or $C$ or $R$ (random)
28:         if road_location = $L$ then
29:             $X_L \leftarrow$ a randomly selected road datum
30:             $X_C \leftarrow$ a randomly selected non-road datum
31:             $X_R \leftarrow$ a randomly selected non-road datum
32:         else if road_location = $C$ then
33:             $X_L \leftarrow$ a randomly selected non-road datum
34:             $X_C \leftarrow$ a randomly selected road datum
35:             $X_R \leftarrow$ a randomly selected non-road datum
36:         else                              ▷ road_location = $R$
37:             $X_L \leftarrow$ a randomly selected non-road datum
38:             $X_C \leftarrow$ a randomly selected non-road datum
39:             $X_R \leftarrow$ a randomly selected road datum
40:         end if
41:     else                                  ▷ number_of_road = 0
42:         $X_L \leftarrow$ a randomly selected non-road datum
43:         $X_C \leftarrow$ a randomly selected non-road datum
44:         $X_R \leftarrow$ a randomly selected non-road datum
45:     end if
46: end if
```

3.2 | Combining information from three views

Given the three inputs, X_L, X_C, and X_R, we fuse them based on the p-guided choice of intersection or non-intersection label. In the next sections, we describe our feature, early, and late fusion methods.

3.2.1 | Feature fusion

With feature fusion, X_L, X_C, and X_R features are combined after global average pooling using a shared-weight feature extraction subnet. Cross entropy loss is used for training, as illustrated in Figure 4A.

3.2.2 | Early fusion

With early fusion, X_L, X_C, and X_R features are concatenated along the channel dimension prior to network input. Because the number of input channels grows three times larger than the original ResNet input in the channel dimension, we enlarge the convolution filter channel size of the first convolution to accommodate the concatenated input, and the convolution filters are randomly initialized. Similar to the feature fusion scheme, this process predicts *intersection* and *non-intersection* classes as illustrated in Figure 4B.

3.2.3 | Late fusion

We train our late fusion method using a *road*/*non-road* classification model using the *road* and *non-road* ground truth as-is. During testing and validation, we feed each X_L, X_C, and X_R to the trained model and post-process its outputs using a voting mechanism. Given two or more *road* class predictions, the final prediction is an *intersection*. Figure 4C presents an illustration of the late fusion method.

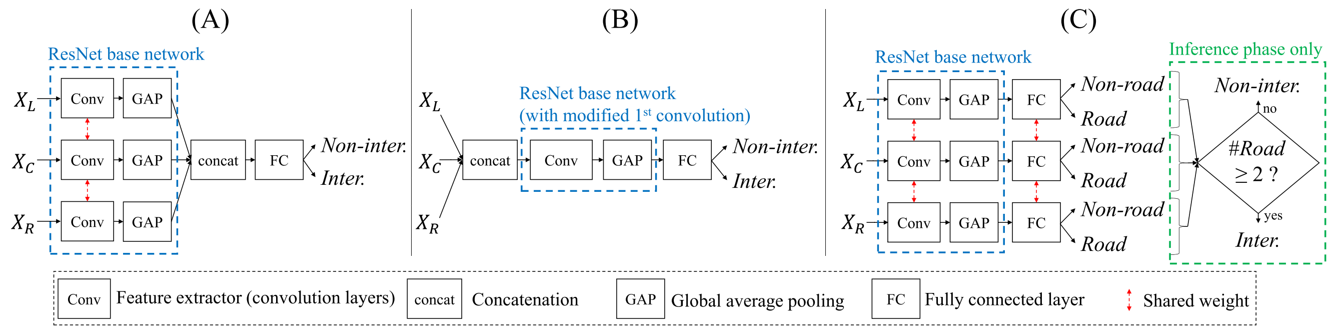

FIGURE 4 Three image fusion methods. (A) feature fusion, (B) early fusion, and (C) late fusion. In feature and early fusion, the model is trained using our proposed technique (Section 3.1). Although tested according to the Figure 2 scenario, the late fusion method was trained with *road* and *non-road* categories.

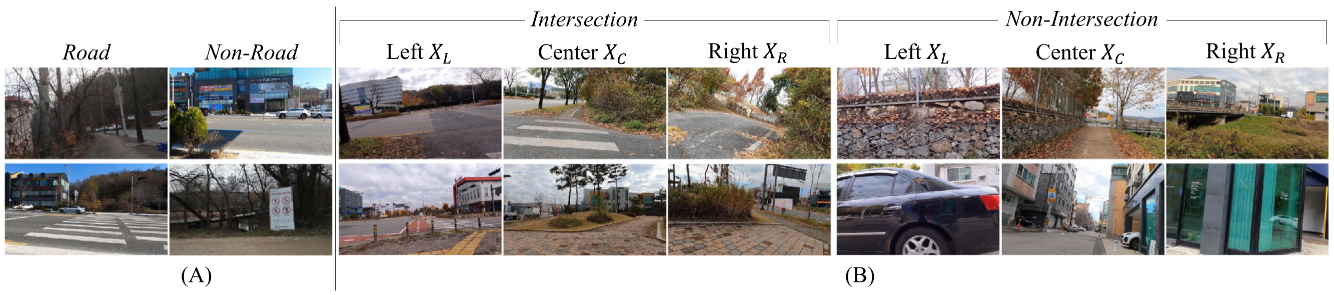

FIGURE 5 Training set image samples collected using (A) one camera (train) and (B) three synchronized cameras (validation and test).

In contrast to feature and early fusion methods, the late-fusion method does not use the proposed augmentation technique described in Section 3.1. Hence, it has the potential disadvantages of class imbalances due to the heavy *road* data proportion; furthermore, it has reduced augmentation opportunities. Additional comparisons are discussed in Section 4.3.1.

4 | EXPERIMENTS

4.1 | Datasets

We conducted our experiments as a pedestrian-view intersection classification problem [1], which is commonly applied to autonomous agent navigation on pedestrian paths. We collected our data in urban areas using phone and action cameras. In theory, the problem addressed by this paper can be adequately addressed with pedestrian-view data, as discussed in Astrid et al. [1]. Moreover, because there are no public pedestrian-view intersection classification datasets, our proposed technique can be beneficial in reducing the cost of data collection and labeling.

The training data are single-camera images collected from phone or action camera. Testing and validation data are three-camera data collected with synchronized three-camera systems (mentioned in Figure 5). In the case of three-camera data, a triplet image is labeled *intersection* if the data were collected in the middle of an intersection.

One-camera data. We collected 568257 *road* and 101386 *non-road* images. The *non-road* data include vehicle roads as seen from sidewalks and dead-ends. The *road* data consist of clear agent paths, including pedestrian crossings. Note that the one-camera data are used only for training. Figure 5A presents some examples.

Three-camera data. For validation and testing, we recorded real three-camera videos and extracted triplet frames corresponding to left, right, and front views. The validation set consists of 11435 triplet frames of 1216 *intersection* and 10219 *non-intersection* types. For the testing set, we collected 81230 triplet frames of 10035 *intersection* and 71195 *non-intersection* types. The goal is to label the triplet as an *intersection* type if the agent is located in the middle of an intersection in reality. Figure 5B presents samples of the collected three-camera data.

4.2 | Experimental setup

4.2.1 | Evaluation metrics

Given *intersection* as positive class and *non-intersection* as negative class, we measure the performance of models using recall, precision, and F1-score. A prediction is considered positive when the confidence of the *intersection* class is greater than 0.5. A low recall value may indicate a high false-negative rate, and a low precision value may denote a high false-positive rate. The overall model performance considering both recall and precision is reflected in the F1 score. Note that accuracy may not be a proper metric when relying on a highly imbalanced validation and testing set.

In binary classification problems, the area under receiver operating characteristic curve (AUC) can be used to measure the separability of positive and negative classes based on the given confidence threshold. A higher AUC value denotes better class separability and assumes a superior binary classifier. Note that AUC cannot be used for the late fusion model because its predictions are based on voting rather than confidence values.

4.2.2 | Implementation details

By default, we train each of our models in an end-to-end manner for 100000 iterations using an Adam optimizer [25] with a mini-batch size of 64, a learning rate of 10^{-5}, and a weight decay of 10^{-6}. Validation is conducted at every 5000 iterations, and we save the best F1 model. At testing time, the saved model is evaluated with the testing set using precision, recall, and F1. AUC is also calculated for the early and feature fusion models. For all metrics, we report the average and standard deviations of five repeated experiments. In addition to the proposed augmentation technique, we apply standard augmentation methods to each image during training, including random contrast, brightness, sharpness, color balance, horizontal flip, and cropping with a scale range of [0.8, 1.0] of the image size. We then resize each frame to 224×224 and normalize its pixel values to the range of [0,1]. During validation and testing, we preprocess the input using only resizing and normalizing. The method was implemented using PyTorch [26], and unless otherwise specified, we used the ImageNet-pretrained ResNet-18 backbone with feature fusion.

4.3 | Quantitative results

4.3.1 | Comparisons of fusion techniques

Table 1 presents comparisons of the three fusion methods described in Section 3.2. Although the early fusion method achieved the highest recall performance, the feature fusion performed the best overall. The early fusion model randomly initializes the first layer, which evidently creates the tendency to falsely identify images as the *intersection* class.

The high performance of the feature fusion method justifies combining information at the feature-level so that the model benefits from the intact pretrained ResNet-18 feature extraction process. As indicated, the late fusion model has disadvantages of being trained with a less balanced training dataset.

4.3.2 | Importance of the ImageNet-pretrained ResNet-18 network

Pretrained models are commonly used to improve overall network performance [27–30]. However, true success depends on the nature of the computer vision task and how well the model fits the task. Hence, in the next experiment, we compared our ImageNet-pretrained model with one trained from a randomly initialized network. As seen in Table 2, fine-tuning a pretrained model leads to higher overall F1 and AUC scores than a from-scratch model. Notably, the latter provided a higher recall but a lower precision value, which seems to indicate that the model tends to produce *intersection* class false positives.

TABLE 1 Comparisons of the three fusion techniques described in Section 3.2.

Fusion	F1	Precision	Recall	AUC
Feature	**46.48 ± 1.44**	**34.63 ± 1.72**	70.95 ± 4.00	**82.00 ± 1.13**
Early	31.48 ± 0.63	19.54 ± 0.60	**81.28 ± 4.88**	73.93 ± 1.48
Late	43.79 ± 1.31	30.37 ± 1.88	79.19 ± 1.13	-

Note: Feature-level fusion works better than the other two fusion methods. Values in bold refer to the best performance for each metric.

TABLE 2 Comparisons between ImageNet- and from-scratch-pretrained networks.

Pretrained	F1	Precision	Recall	AUC
Yes	**46.48** ± **1.44**	**34.63** ± **1.72**	70.95 ± 4.00	**82.00** ± **1.13**
No	36.02 ± 0.56	23.36 ± 0.63	**78.84** ± **3.26**	77.70 ± 0.55

Note: Values in bold refer to the best performance for each metric.

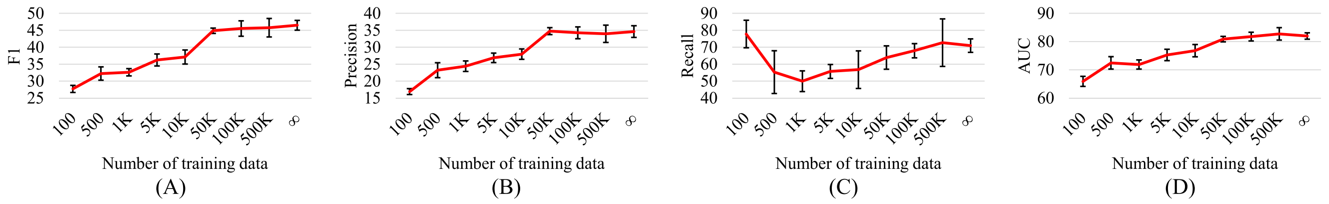

FIGURE 6 We built several three-camera intersection classification datasets with different fixed numbers of triplet frames for training. "Unlimited" refers to our one-camera augmentation method, which can theoretically produce k^3 new training data from the original single-camera images. The average and standard deviation of (A) F1, (B) precision, (C) recall, and (D) AUC are reported. As can be seen, increasing the number of training data generally improves intersection classification performance.

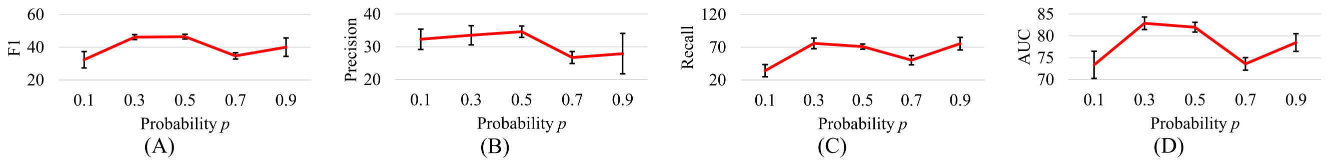

FIGURE 7 Our model's performance trained with different class-balancing hyperparameter p. The higher the p, the greater is the *intersection* bias and vice versa. The averages and standard deviations of (A) F1, (B) precision, (C) recall, and (D) AUC are reported, reflecting the importance of having a balanced dataset.

4.3.3 | Number of training data

As mentioned in Section 3.1, our proposed augmentation technique enables us to obtain vast amounts of training samples without having to collect new real-world data. With $k = 568257 + 101386 = 669643$ (one-camera data), we can produce more than 3×10^{17} three-camera training data, giving us practically an infinite number of data; hence, the ∞ symbol in Figure 6.

To see how the number of training data affects performance, we built several three-camera datasets of various sizes, between 100 and 500K data items, using augmentation techniques. Figure 6 shows the increasing trend of ImageNet-pretrained ResNet-18 feature fusion performance as we increased the number of training data. As seen, with more training data, the overall F1 and AUC performance improves, plateauing at around 100 000 data, likely due to the model's capacity. Conversely, we did not have to manually collect another 100 000 balanced three-camera data items. Although the recall value was very high for the 100 training data, the precision was very low, which led to a lower overall performance. This indicates that the 100 data model has a tendency to predict most of the test data as *intersection*.

4.3.4 | Balancing the dataset

Our novel class-balancing method was highly advantageous to our model's improved performance. As described in Section 3.1 and Algorithm 1, we can set the p value to different real numbers in [0, 1], where 1 is producing all intersection data and no non-intersection at all. Notably, as seen in Figure 7, reducing $p = 0.5$ to around $p = 0.3$ continues to provide a good data balance for our three-camera intersection classification model. However, with lower p values (such as $p = 0.1$), recall suffers considerably owing to the resultant higher false negative rate. With greater p values ($p \geq 0.7$), false positives increase, leading to lower precision. This results in a higher false positive that lowers precision. Note that it is extremely difficult to collect a sufficiently large set of real and balanced data to train a deep model; however, our approach can adjust the data balance freely by setting p to an appropriate value.

4.3.5 | Augmentation algorithm comparisons

Given the augmentation technique described in Algorithm 1, there is another way to sample *road* and *non-road* datasets to create three-camera intersection classification data. In this section, we sample the eight cases shown in Figure 2, each with an equal probability. Hence, overall, p still equals 50%. As shown in Table 3, both are effective in creating three-camera intersection classification datasets from one-camera data.

4.3.6 | Augmented versus real three-camera data

To demonstrate the effectiveness of data augmentation, we compared our model with a baseline trained on real three-camera data. Owing to the limited number of data that were available, we used the validation set for training and the testing set for validation on both models. We already knew that the baseline model would be more likely to overfit to the testing set. Nevertheless, as seen in Table 4, the comparison model had higher precision, but it produced more false negatives, leading to low recall. This translated to an overall worse F1 and AUC than our proposed method.

4.3.7 | Learning rate evaluations

Figure 8 shows the F1 and AUC performances of our model trained with various learning rates. It shows that the performance of our model is impaired when the learning rate is too small.

4.3.8 | Batch size evaluations

Figure 9 shows the overall F1 and AUC scores of our model under different batch-size training settings. When the batch size was too large, overfitting occurs [31], particularly if the training data are not real data.

4.3.9 | Combining with MixUp

MixUp [22] is a widely-used augmentation technique that linearly combines two images at a randomly selected ratio. We applied MixUp to *road*/*non-road* data to

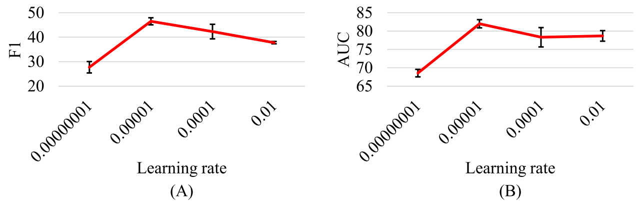

FIGURE 8 Averages and standard deviations of (A) F1 and (B) AUC scores for our model trained with different learning rate settings. Apart from the case when the rate was too low, our model is robust to a wide range of rates.

TABLE 3 Augmentation algorithm comparisons between uniform sampling from the eight cases illustrated in Figure 2 versus using our Algorithm 1.

Algorithm	F1	Precision	Recall	AUC
Uniform 8 cases	46.28 ± 0.98	33.49 ± 1.98	**75.75 ± 7.12**	**82.92 ± 1.36**
Algorithm 1	**46.48 ± 1.44**	**34.63 ± 1.72**	70.95 ± 4.00	82.00 ± 1.13

Note: Both methods are equally effective in creating three-camera data from one-camera images. Values in bold refer to the best performance for each metric.

TABLE 4 Comparisons between our model trained with the augmented dataset built from one-camera data versus the baseline trained with real three-camera validation data.

Training data	F1	Precision	Recall	AUC
Augmented	**46.48 ± 1.44**	34.63 ± 1.72	**70.95 ± 4.00**	**82.00 ± 1.13**
Real	6.07 ± 1.82	**52.60 ± 6.10**	3.24 ± 0.90	70.82 ± 1.82

Note: Owing to severe data imbalances and a fewer number of images, training with real data is clearly suboptimal. Values in bold refer to the best performance for each metric.

generate new X_R, X_C, and X_L according to randomly selected ratios of *road* data, r_R, r_C, and r_L, respectively. Intuitively, the larger the *road* data proportion, the more likely the triplet data will be judged an *intersection*. Therefore, we calculated the final *intersection*/*non-intersection* labels in two ways:

- Average (avg): $y = (r_R + r_C + r_L)/3$
- Summation (sum):

$$y = \begin{cases} 0 & \text{if } y_{\text{sum}} <= 1, \\ y_{\text{sum}} - 1 & \text{if } 1 < y_{\text{sum}} < 2, \\ 1 & \text{if } y_{\text{sum}} >= 2, \end{cases} \quad (1)$$

where $y_{\text{sum}} = r_R + r_C + r_L$.

Table 5 shows the performance comparisons of our method with/without MixUp. As seen, although MixUp was effective for generic image classification tasks, it was not as effective (or much worse) when attempting to handle three-camera intersection classification tasks. This was probably caused by the data distribution of MixUp being too intuitively distant from the test data. Conversely, our augmented dataset without MixUp provided a closer distribution to the real data, which led to a better overall performance.

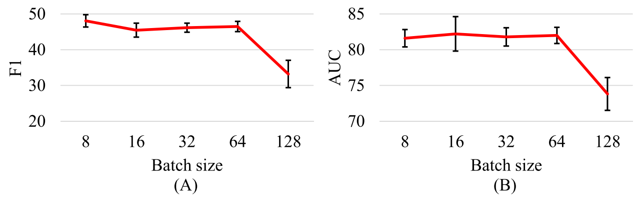

FIGURE 9 Averages and standard deviations of (A) F1 and (B) AUC of our model trained on various batch-size settings. Our model is generally robust for small batch sizes until the setting reaches 64.

4.3.10 | Comparisons with a one-camera model

To provide a fair comparison with single-camera approaches, we choose a pedestrian-view intersection classification model from Astrid et al. [1] and tested it using our dataset. In Astrid et al. [1], an *intersection* is tagged when it is nearby and clearly seen by the front camera. This comparison model was the most similar to ours, and it also used ResNet-18 (for pedestrian-view intersection classification). Moreover, to cope with cases where an *intersection* is defined as the *middle* of the intersection, we also trained a one-camera model on the X_C of the real three-camera data taken from the validation set.

The results are listed in Table 6. Compared with our three-camera model, the significantly lower precision values of the model provided by Astrid et al. [1] shows that it detects many intersections before their middle (Figure 1A), whereas the lower recall value shows that it misses more intersections in their middle (Figure 1B). The low performance of the one-camera model trained with the validation set can be attributed to the confusion generated when separating intersection and non-intersection classes during training (Figure 1C). The rationale for this deficiency was explained earlier.

TABLE 5 Comparisons between our model with/without MixUp. The data distribution provided by the MixUp algorithm too intuitively distant from the test data, which led to lower performance.

Model	F1	Precision	Recall	AUC
Ours	**46.48 ± 1.44**	**34.63 ± 1.72**	**70.95 ± 4.00**	**82.00 ± 1.13**
+ MixUp (avg)	33.83 ± 2.67	24.79 ± 1.07	53.82 ± 9.24	74.01 ± 2.23
+ MixUp (sum)	35.64 ± 0.02	26.47 ± 2.62	56.33 ± 11.79	74.65 ± 0.21

Note: Values in bold refer to the best performance for each metric.

TABLE 6 Comparisons between our three-camera model, the one-camera model from Astrid et al. [1], and the one-camera model trained on our real three-camera data (validation set) tested on our testing set.

Model	F1	Precision	Recall	AUC
3-camera	**46.48 ± 1.44**	**34.63 ± 1.72**	**70.95 ± 4.00**	**82.00 ± 1.13**
1-camera (Astrid et al. [1])	26.50 ± 0.76	17.56 ± 0.10	54.31 ± 5.41	62.81 ± 0.84
1-camera (val)	9.11 ± 2.55	32.97 ± 3.22	5.31 ± 1.64	59.62 ± 1.19

Note: The results show the superiority of our three-camera model compared with those of the others in detecting whether an agent is in the middle of an intersection. Values in bold refer to the best performance for each metric.

TABLE 7 Comparisons of different ResNet depths as the base network.

Base network	F1	Precision	Recall	AUC
ResNet-18	46.48 ± 1.44	34.63 ± 1.72	70.95 ± 4.00	82.00 ± 1.13
ResNet-34	47.38 ± 1.31	36.30 ± 3.16	70.61 ± 12.20	**82.48 ± 1.99**
ResNet-50	**47.70 ± 2.24**	**36.31 ± 3.28**	**71.09 ± 9.11**	82.13 ± 1.73

Note: We tested on ResNet-18, ResNet-34, and ResNet-50, ordered from shallowest to deepest. As seen, deeper networks improve performance. Values in bold refer to the best performance for each metric.

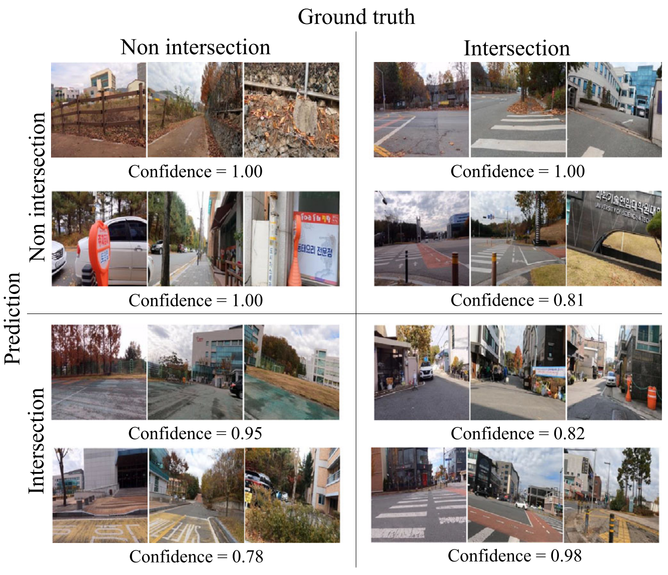

FIGURE 10 Examples of true negatives (top-left), false negatives (top-right), false positives (bottom-left), and true positives (bottom-right) results produced by our model. Each three-camera datum is given as a left–center–right triplet. Detailed descriptions are provided in Section 4.4.

4.3.11 | Increasing network capacity

To improve network capacity with respect to the training data and improved feature extractor, we can apply a deeper network as the backbone. As shown in Table 7, we compared our approach to different ResNet model depths, from the shallowest ResNet-18 to the deepest ResNet-50. We found that using deeper networks can improve performance, which provides considerable promise in terms of capability improvement.

4.4 | Qualitative results

The qualitative results generated by our model can be seen in Figure 10. Our model correctly identified *intersection* and *non-intersection* in diverse scenes. False positive cases still occur when the data are ambiguous. Additionally, our model produced some false negatives in some crosswalk cases. This is likely due to the lack of crosswalk data in the training dataset.

5 | CONCLUSION

Detecting whether an agent is inside an intersection is important for autonomous navigation tasks. To improve upon this capability, we proposed a three-camera intersection classification model trained on left–front–right camera views. To overcome the difficulties of producing three-camera datasets from real-world conditions, we utilized a small number of publicly available one-camera data, and built a three-camera intersection classifier model. To use one-camera data for three-camera classification, an augmentation technique and three different approaches for combining the information from three cameras were provided. From extensive experiments, we found that combining feature information achieves the best performance over other fusion methods. We also demonstrated the importance of increasing the quantity of training data, as long as the resulting dataset is balanced. Our augmentation method ensures this. Finally, owing to the considerable amount of data, we safely increased the depth of the network without increasing overfitting, which led to performance improvements. Therefore, our approach has considerable potential. Future works should explore the effects of additional fusion methods and the possibility of generating 360-degree scene data from one-camera images. Moreover, our approach can be adaptable for autonomous vehicle tasks where new environments are expected to be encountered. For example, self-navigating mining robots would benefit greatly from this capability. Moreover, planetary and moon rovers can benefit from *non-road* classifications of crater, cliff, boulder, and other obstacles.

CONFLICT OF INTEREST STATEMENT

The authors declare that there are no conflicts of interest.

ORCID

Marcella Astrid https://orcid.org/0000-0003-1432-6661
Seung-Ik Lee https://orcid.org/0000-0003-2986-7540

REFERENCES

1. M. Astrid, J.-H. Lee, M. Z. Zaheer, J.-Y. Lee, and S.-I. Lee, *For safer navigation: pedestrian-view intersection classification*, (International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Republic of Korea), 2020, pp. 7–10.
2. A. L. Ballardini, A. H. Saz, and M. A. Sotelo, *Model guided road intersection classification*, (IEEE Intelligent Vehicles Symposium (iv), Nagoya, Japan), 2021, pp. 703–709.
3. S. Kumaar, B. Navaneethakrishnan, S. Mannar, and S. N. Omkar, *JuncNet: a deep neural network for road junction disambiguation for autonomous vehicles*, arXiv preprint, 2018, arXiv:1809.01011.
4. S. S. Mansouri, P. Karvelis, C. Kanellakis, A. Koval, and G. Nikolakopoulos, *Visual subterranean junction recognition for MAVS based on convolutional neural networks*, (IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society, Lisbon, Portugal), 2019, pp. 192–197.
5. V. Tümen and B. Ergen, *Intersections and crosswalk detection using deep learning and image processing techniques*, Phys A: Stat. Mech. Appl. **543** (2020), 123510.
6. T. Watanabe, K. Matsutani, M. Adachi, T. Oki, and R. Miyamoto, *Feasibility study of intersection detection and recognition using a single shot image for robot navigation*, J. Image Graph. **9** (2021), no. 2, 39–44.
7. M. Astrid, M. Z. Zaheer, J.-Y. Lee, and S.-I. Lee, *Domain-robust pedestrian-view intersection classification*, (International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Republic of Korea), 2021, pp. 1087–1090.
8. D. Bhatt, D. Sodhi, A. Pal, V. Balasubramanian, and M. Krishna, *Have I reached the intersection: a deep learning-based approach for intersection detection from monocular cameras*, (2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, Canada), 2017, pp. 4495–4500.
9. A. Fathi and J. Krumm, *Detecting road intersections from GPS traces*, *International Conference on Geographic Information Science*, Springer, 2010, pp. 56–69.
10. A. Mueller, M. Himmelsbach, T. Luettel, F. Hundelshausen, and H.-J. Wuensche, *GIS-based topological robot localization through LIDAR crossroad detection*, (14th International IEEE Conference on Intelligent Transportation Systems (ITSC), Washington, DC, USA), 2011, pp. 2001–2008.
11. Q. Zhu, L. Chen, Q. Li, M. Li, A. Nüchter, and J. Wang, *3D LIDAR point cloud based intersection recognition for autonomous driving*, (IEEE Intelligent Vehicles Symposium, Madrid, Spain), 2012, pp. 456–461.
12. U. Baumann, Y.-Y. Huang, C. Gläser, M. Herman, H. Banzhaf, and J. M. Zöllner, *Classifying road intersections using transfer-learning on a deep neural network*, (21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA), 2018, pp. 683–690.
13. T. Koji and T. Kanji, *Deep intersection classification using first and third person views*, (2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France), 2019, pp. 454–459.
14. P. Mukhija, S. Tourani, and K. M. Krishna, *Outdoor intersection detection for autonomous exploration*, (15th International IEEE Conference on Intelligent Transportation Systems, Anchorage, AK, USA), 2012, pp. 218–223.
15. Y. Nie, Q. Chen, T. Chen, Z. Sun, and B. Dai, *Camera and LIDAR fusion for road intersection detection*, (IEEE Symposium on Electrical & Electronics Engineering (EEESYM), Kuala Lumpur, Malaysia), 2012, pp. 273–276.
16. A. Seff and J. Xiao, *Learning from maps: visual common sense for autonomous driving*, arXiv preprint, 2016, arXiv:1611.08583.
17. D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver, *Google Street View: capturing the world at street level*, Computer **43** (2010), no. 6, 32–38.
18. Y. Bengio, F. Bastien, A. Bergeron, N. Boulanger-Lewandowski, T. Breuel, Y. Chherawala, M. Cisse, M. Côté, D. Erhan, and J. Eustache, *Deep learners benefit more from out-of-distribution examples*, (Proceedings of the 14th International Conference on Artificial Intelligence and Statistics), 2011, pp. 164–172.
19. A. Krizhevsky, I. Sutskever, and G. E. Hinton, *ImageNet classification with deep convolutional neural networks*, (Advances in Neural Information Processing Systems), 2012, pp. 1097–1105.
20. A. Dabouei, S. Soleymani, F. Taherkhani, and N. M. Nasrabadi, *SuperMix: supervising the mixing data augmentation*, (Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA), 2021, pp. 13794–13803.
21. J.-H. Lee, M. Z. Zaheer, M. Astrid, and S.-I. Lee, *SmoothMix: a simple yet effective data augmentation to train robust classifiers*, (Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA) 2020, pp. 756–757.
22. H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, *mixup: beyond empirical risk minimization*, (International Conference on Learning Representations), 2018.
23. S. J. Pan and Q. Yang, *A survey on transfer learning*, IEEE Trans. Knowledge Data Eng. **22** (2009), no. 10, 1345–1359.
24. K. Weiss, T. M. Khoshgoftaar, and D. Wang, *A survey of transfer learning*, J. Big Data **3** (2016), no. 1, 1–40.
25. D. P. Kingma and J. Ba, *Adam: a method for stochastic optimization*, arXive preprint, 2014, arXiv:1412.6980.
26. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, *Pytorch: an imperative style, high-performance deep learning library*, *Advances in neural information processing systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, (eds.), Curran Associates, Inc., 2019, pp. 8024–8035.
27. Y. Kim, D. Ra, and S. Lim, *Zero-anaphora resolution in Korean based on deep language representation model: Bert*, ETRI J. **43** (2021), no. 2, 299–312.
28. C. Park, *Multi-task learning with contextual hierarchical attention for Korean coreference resolution*, ETRI J. **45** (2023), no. 1, 93–104.
29. A. K. Reyes, J. C. Caicedo, and J. E. Camargo, *Fine-tuning deep convolutional networks for plant recognition*, CLEF (Working Notes) **1391** (2015), 467–475.

30. J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, *How transferable are features in deep neural networks?* (Proc. International Conference on Neural Information Processing Systems, Montreal, Canada), 2014, pp. 3320–3328.
31. N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, *On large-batch training for deep learning: generalization gap and sharp minima*, (International Conference on Learning Representations), 2017.

AUTHOR BIOGRAPHIES

Marcella Astrid received her BEng in computer engineering from the Multimedia Nusantara University in Tangerang, Indonesia in 2015. She received her MEng in computer software and the PhD in artificial intelligence from the University of Science and Technology (UST) in Daejeon, Korea, in 2017 and 2023, respectively. She is currently associated with the University of Luxembourg as a postdoctoral researcher. The research presented in this paper was largely carried out during her PhD work, and revisions were made at her current affiliation. Her research interests include one-class classifications, anomaly detection, and deepfake detection.

Seung-Ik Lee received his BS, MS, and PhD degrees in computer science from Yonsei University in Seoul, Korea, in 1995, 1997, and 2001, respectively. He currently works at the Electronics and Telecommunications Research Institute in Daejeon, Korea. Since 2005, he has been with the Department of Artificial Intelligence at the University of Science and Technology in Daejeon, Korea, where he is a professor. His research interests include computer vision, anomaly detection, object detection, open-set learning, out-of-distribution data detection, domain adaptation, learning with limited data, deep learning, and reinforcement learning.

How to cite this article: M. Astrid and S.-I. Lee, *Assembling three one-camera images for three-camera intersection classification*, ETRI Journal **45** (2023), 862–873. DOI 10.4218/etrij.2023-0100