Auto Annotation using Object Tracking with Multiple in-vehicle Cameras for Federated Learning

Yuuki Nakahama, Satoshi Ohzahata, Ryo Yamamoto
Graduate School of Informatics and Engineering
The University of Electro-Communications, Tokyo, Japan
y.nakahama@net.lab.uec.ac.jp, ohzahata@uec.ac.jp, ryo-yamamoto@uec.ac.jp

***Abstract*—In recent years, significant advancements have been made in automated driving technology. To achieve fully automated driving, a vehicle must accurately acquire information regarding objects in its surroundings by gathering large amounts of data for machine learning. In many cases, data collected from each vehicle is transmitted to a central server, which is undesirable because of privacy concerns and traffic generation to retrieve the data. Federated Learning (FL) is a promising technology that addresses these concerns by enabling learning while maintaining the data with the user. However, since the local labeling accuracy for distant objects with existing models is still low, quality of the model training is affected. To address this issue, this paper proposes the use of Multiple object Tracking (MoT) with multiple cameras to improve the labeling accuracy for FL. Our method enables the application of the labeling results of nearby objects to identify distant objects by tracking time-series data from multiple cameras. The evaluation results demonstrate that the proposed method effectively labels distant objects without sending data to the central server.**

***Index Terms*—Object Tracking, Labeling, Federated Learning**

I. Introduction

The development of automated driving technologies has been remarkable. To achieve fully automated driving, it is crucial to accurately acquire information about the objects around a vehicle, and machine learning is often used. To realize accurate object recognition with machine learning, a large amount of data is required, and a central server generally creates a model by collecting data [1]. However, collecting data generated by each vehicle to the central server is often undesirable from the perspective of privacy and data collection traffic.

Federated Learning (FL) has gained attention because of its ability to enable each user to update a machine learning model without transferring their private data to the central server [2]. In this procedure, a global model is created by the central server and distributed to each client. Each client then creates a local model by learning from its data and sends the difference in parameters between the local and global models to the central server. The server subsequently creates a new global model incorporating the information received from the clients and distributes the updated model to each client. Through this process, the model is improved while maintaining privacy by eliminating the need for clients to send their private data to the central server.

In addition, annotation must be performed manually or automatically on the obtained data before starting the training process for supervised learning [3], [4]. However, since FL does not allow data to be sent to the central server, the annotation must also be performed locally. Therefore, three problems must be solved to perform annotation for the object recognition in the FL. The first problem is that manual labeling is not feasible because of the high cost and lack of labelers in the local environment [5]. The second problem is the accuracy of the object recognition model. Although auto-annotation is preferred, the accuracy of the current model is insufficient for distant objects [6]. The third problem is occlusions. Some objects in the images captured by the camera can have reduced detection accuracy because of overlapping, which requires additional information from different angles [7].

In this paper, we propose to realize auto annotation for FL and improve the labeling accuracy to solve the first and second problems by using Multiple object Tracking (MoT) to label distant objects that cannot be labeled accurately by object recognition in a local environment. We also use multiple cameras to solve the third problem. We use data generated by CARLA simulator [8] to evaluate the effectiveness of the proposed method.

II. Related Techniques

This section discusses related technologies as part of our proposed method.

A. Object detection

You Look Only Once (YOLO) is a real-time object detection tool that labels objects in an image and calculates their confidence level [9]. As shown in Figure 1, labeling in this paper refers to defining a bounding box that indicates where the object exists and a class that indicates what the object is, so that we can determine where and what kind of object exists in the image. The confidence level is a score that indicates how accurate the labeling values are and is calculated from "how accurately each bounding box contains an object and how accurately the region is surrounded" and "the predicted probability of each class."

Fig. 1. Labeling.

B. Multiple object Tracking (MoT)

Multiple object Tracking (MoT) is a technique for identifying the same object in sequential images by tracking their movement. We used motpy [10] that performs object tracking using the Kalman filter. Based on the information of the current image, the object is estimated in the next image, and the detected objects are compared to track them. In this paper, MoT is performed on time-series images acquired from a camera installed in a vehicle to determine whether a distant object is the same object when the object approaches the vehicle.

C. Mapping 2D objects to 3D object

PointPainting [11] is an improved object detection method that maps 3D objects by extending the dimensions of point cloud data obtained from LiDAR with scores obtained from the semantic segmentation of 2D images. First, a segmentation score is calculated for each pixel by performing semantic segmentation on the 2D camera image. Next, the scores are mapped to the point cloud data obtained from Lidar to expand the dimension of the data. Finally, 3D object detection is performed using the extended point cloud. Because an object detected by cameras can be mapped to the same 3D object detected by LiDAR, we can identify the same object in different cameras.

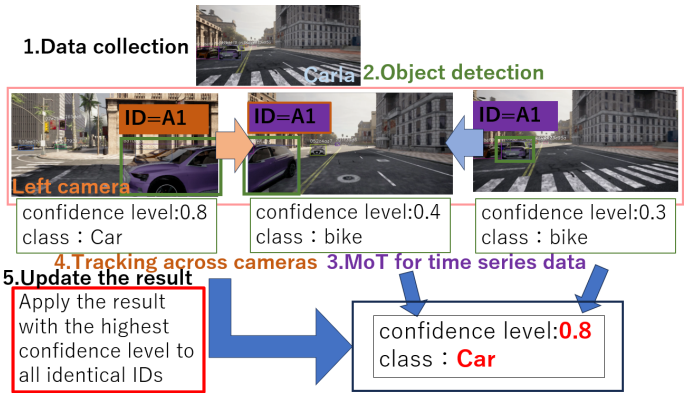

Fig. 2. Proposed method procedure.

III. Proposed Method

To realize FL in actual data collection procedures, we propose an auto annotation method with Multiple object Tracking (MoT) using multiple cameras. In general, object detection methods do not enable the identification of distant objects with high accuracy. However, if the object exists near the camera, it can be detected with high accuracy. Thus, we apply the detection result of high accuracy to the object when the object is away from the camera if both objects are identical. To determine whether both objects are identical, we used the MoT. In addition, identifying the same object across different cameras, we track an object using LiDAR information when the object moves between cameras. By mapping objects from different cameras to the same object in LiDAR, we can accurately identify and track objects across multiple cameras. Figure 2 shows the procedure of the proposed method. "1. data collection by cameras," "2. object detection" is performed to assign labels and the confidence levels to all the detected objects. After that "3. MoT for time-series data" generated by cameras is done, and then "4. Tracking across different cameras" assigns the same ID to the same object by using the identified 3D object. "5. Update the result" is to update the label by the label of the highest confidence level to the objects identified as the same one. The details are as follows:

1. Data Collection

We collect image data from the Carla simulator [8] [12] and use a vehicle equipped with cameras and LiDAR. The vehicle moves on an urban map to capture data periodically. During the data collection, correct labels for objects in the image data are also obtained from the simulator environment to evaluate the labeling accuracy.

2. Object Detection

We use YOLO to assign labels and confidence levels to objects that exist in all the collected image data of the in-vehicle cameras. Using this procedure, we can obtain the object detection result with the highest confidence level in the time-series data.

3. MoT for time-series data

The proposed method uses MoT to assign the same ID to the same object in the acquired time-series data. The pseudo-code for this process is shown in Algorithm 1, and an example of object tracking is illustrated in Figure 3. By tracking the object, we can identify that the distant object in the left image is the same as the nearby vehicle in the right image. This procedure enables the assignment of the same ID to a distant object with the highest confidence level obtained at a close range of the camera.

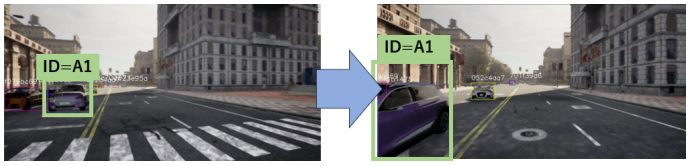

Fig. 3. Tracking.

4. Tracking across cameras

The proposed method also assigns the same ID to objects that exist simultaneously in multiple cameras. Figure 4 shows an example of an object that exists across the cameras. The pseudo-code for this procedure is shown in Algorithm 2. First,

Algorithm 1 MoT(Multiple object Tracking): Explain with front camera image.

$ImageDataF \leftarrow [\ ]$
$NumImage$ //Number of images to be used
$num1 \leftarrow 0$
for $NumImage$ **do**
  $labels, confidence \leftarrow YOLOX(Image[num1])$ //Object detection by YOLOX
  $ObjectData \leftarrow [\ ]$
  $NumObject \leftarrow numberofobject$ //Number of objects detected by YOLOX
  $num2 \leftarrow 0$
  for $NumObject$ **do**
    $ID \leftarrow MoT(labels[num2], Image[num1], labels\_old, Image\_old)$ //Object Tracking with time-series data
    $ObjectData$ APPEND((ID, $labels[num2]$, $confidence[num2]$))
    $num2 \leftarrow num2 + 1$
  end for
  $ImageDataF$ APPEND($ObjectData$)
  $num1 \leftarrow num1 + 1$
end for

Algorithm 2 Tracking across cameras: Explain with data from front and left cameras.

$SameObjectFL \leftarrow [\ ]$ //Location to store object IDs that are determined to be identical
$NumImage$ //Number of images to be used
$num1 \leftarrow 0$
for $NumImage$ **do**
  $3DcoodinatesF, 3DbboxsF \leftarrow PointPainting(ImageF, LiDARF)$ //Calculate 3D coordinates and 3D bounding box from frontal data
  $3DcoodinatesL, 3DbboxsL \leftarrow PointPainting(ImageL, LiDARL)$
  $Num3DobjectF \leftarrow numberof3DbboxsF$ //Number of objects detected by PointPainting from frontal data
  $num2 \leftarrow 0$
  for $Num3DobjectF$ **do**
    $Num3DobjectL \leftarrow numberof3DbboxsL$
    $num3 \leftarrow 0$
    for $Num3DobjectL$ **do**
      //Compare 3D bounding boxes in the same location
      if $3DcoodinatesF[num2]$ == $3DcoodinatesL[num3]$ **then**
        $Num2DobjectF$ //Get the number of ImageDataF[num1] from the result of Algorithm1
        $num4 \leftarrow 0$
        for $Num2DobjectF$ **do**
          if $3DbboxsF[num2]$ IN $ImageDataF[num1][num4]$ **then**
            $Num2DobjectL \leftarrow numberofImageDataL[num1]$
            $num5 \leftarrow 0$
            for $Num2DobjectL$ **do**
              //Compare whether each bounding box exists in the same location
              if $3DbboxsL[num3]$ IN $ImageDataL[num1][num5]$ **then**
                $SameObjectFL$ APPEND($ImageDataF[num1][num4][ID]$,$ImageDataL[num1][num5][ID]$) //Save each ID of the same object
              end if
            end for
          end if
          $num4 \leftarrow num4 + 1$
        end for
      end if
      $num3 \leftarrow num3 + 1$
    end for
    $num2 \leftarrow num2 + 1$
  end for
  $num1 \leftarrow num1 + 1$
end for

Algorithm 3 Update Results: Explain with data from front and left cameras.

{Get the ID with the highest confidence level and its confidence level and class from the MoT results}
$HightScoreF$ = [ ] //Location to store the ID, confidence level, and class
$NumImage$ //Number of images to be used
$num1 \leftarrow 0$
for NumImage **do**
  $NumObjectF$ //Get the number of ImageDataF[num1] from the result of Algorithm1
  $num2 \leftarrow 0$
  for NumObjectF **do**
    //Determine if it exists within the $HightScoreF$
    if NoT ($ImageDataF[num1][num2][ID]$ IN $HightScoreF$) **then**
      $HightScoreF$ APPEND($ImageDataF[num1][num2][ID]$: $ImageDataF[num1][num2][confidence]$, $ImageDataF[num1][num2][label][class]$)
      //Compare the confidence level of stored objects with the same ID
    else if $ImageDataF[num1][num2][confidence] > HightScoreF[ImageDataF[num1][num2][ID]][confidence]$ **then**
      //Update HightScoreF
      $HightScoreF[ImageDataF[num1][num2][ID]][label][class] \leftarrow ImageDataF[num1][num2][label][class]$
      $HightScoreF[ImageDataF[num1][num2][ID]][confidence] \leftarrow ImageDataF[num1][num2][confidence]$
    end if
    $num2 \leftarrow num2 + 1$
  end for
  $num1 \leftarrow num1 + 1$
end for
$NumHightScore$ //Number of elements in HightScoreF
$num3 \leftarrow 0$
for $NumElementHight$ **do**
  $NumSameobjectFL$ //Get the number of SameObjectFL from the result of Algorithm2
  $num4 \leftarrow 0$
  for $NumSameobjectFL$ **do**
    if $HightScoreF[num3][ID]$ == $SameObjectFL[num4][FrontID]$ **then**
      if $HightScoreF[num3][confidence] < HightScoreL[SameObjectFL[num4][LeftID]][confidence]$ **then**
        //Update HightScoreF
        $HightScoreF[num3][label][class] \leftarrow HightScoreL[SameObjectFL[num4][LeftID]][label][class]$
        $HightScoreF[num3][confidence] \leftarrow HightScore[SameObjectFL[num4][LeftID]][confidence]$
      end if
    end if
    $num4 \leftarrow num4 + 1$
  end for
  $num3 \leftarrow num3 + 1$
end for

PointPainting simultaneously obtains the 3D coordinates of each object in the image acquired from each camera. Then, by comparing the 3D coordinates obtained from the left camera data with those obtained from the front camera data (Figure 4), we determine whether the objects exist in the same location. If they exist in the same location, the objects are identical because they cannot exist in overlapping locations.
Next, we determine whether the object detected by PointPainting and the object detected by YOLO are the same by comparing the position of the 3D bounding box of PointPainting and the position of the 2D bounding box of YOLO. We can then assign the same ID to the same object existing in different cameras by referring to the object determined to have the same coordinates in PointPainting and in YOLO.

5. Update the results of object detection

To update the class of all data with the class of the object with the highest confidence level, we compare the confidence levels of the objects with the same ID. Because the same ID is given to the same object by "3. MoT of time series data" and "4. Tracking across cameras," all objects with the same

Fig. 4. Objects straddling between cameras.

ID are considered to be in the same class. As the result for the object with the highest confidence level is the most accurate labeling result, this result can be applied to all objects with the same ID to change the class of all the objects with the same ID to the appropriate one.

This procedure is performed for all IDs in the acquired data, as shown in Algorithm 3. In Figure 2, for example, among the objects with ID A1, the object on the left image has the highest confidence level of 0.8. By applying this result to objects with the same ID in the center and right images, which also have ID A1, the respective class values can be updated. This approach corrects mislabeling and improves the accuracy of object detection in multiple-camera environments.

IV. Evaluation Experiments

A. Evaluation setups and metrics

The experimental environment is listed in Table I. To evaluate the proposed method, we used the automated driving simulator Carla 0.9.12 and set up it for data acquisition as shown in Table II. "The detection distance" is used to determine the distance that an object must be detected from a vehicle. We compare the results of the proposed method using a single camera with and without an MoT.

As an evaluation metric in object detection, we use Intersection over Union (IoU), which represents the degree of overlap between images, as shown in Figure 5. The IoU is calculated as the overlap between the bounding box obtained by object detection and the correct bounding box of the data. In this metric, the larger the overlap between the bounding boxes, the larger is the IoU value. If the IoU value exceeds the threshold, we define that the position of the object is predicted correctly. We change the IoU threshold for object detection with 0.25 and 0.50 as the evaluation condition.

TABLE I
EXPERIMENT ENVIRONMENT.

Item	Setting condition
Learning model	YOLOX-X [9]
MoT method	motpy [10]
3D object detection model	PointPainting [13]

To evaluate the object detection accuracy, we use the following evaluation metrics:

- Precision: is expressed by the following equation: $Precision = \frac{TP}{TP+FP}$
- Recall: is expressed by the following equation: $Recall = \frac{TP}{TP+FN}$

TABLE II
CARLA'S SETTING.

Item	Setting condition
Use map	Town10
weather	Clear
Generated object	Vehicles only
Number of vehicles	60 vehicles
Number of vehicles with cameras installed	1 vehicle
Shooting interval	0.1 second
Image resolution	720 × 360
Number of data acquired	1000
Detection distance	80m
Driving path	Random turn at intersection
Camera position	Roof of a car
Camera direction	Front, Rear, Left, Right

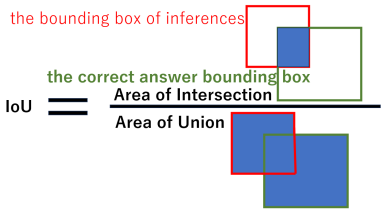

Fig. 5. Intersection over Union (IoU).

- True Positive (TP): Objects of the detected bounding box class are correct, and whose the IoU value is higher than the threshold.
- False Positive (FP): Detected objects whose class are incorrect, or detected object class is correct but IoU value is less than the threshold value.
- False Negative (FN): Objects are not detected.

B. Experimental results

Table III shows the detection results. In general, training data should not be created with incorrect results. Therefore, we set the confidence level sufficiently high and set it to 0.8. The proposed method utilizes a 4-camera MoT in that cameras are placed at the front, both sides, and rear. The MoT using only the front camera is a single-camera MoT. "Without MoT" is the results obtained when only object detection is performed for each image. In the results, since both the Precision and Recall of MoT are higher than those without MoT, we can say that MoT can improve labeling accuracy. The reason for the improvement in Precision is that the "MoT" methods update the results of low-confidence level objects with the results of the highest-confidence level object using the time-series data, which reduces false positives and improves labeling accuracy.

The results for Recall show that the value of Recall with MoT is higher than that of without MoT for both conditions of IoU. This is because MoT enables the use of the highest confidence level of the detection results in the time-series data to distant objects with a low confidence level, which leads to an increase in the number of detected objects, resulting in a decrease in FN, and an increase in TP. These results show that the methods of "MoT" are capable of enhancing the accuracy of object detection by enabling the application of near results to distant objects.

TABLE III
DETECTION RESULTS WITH A CONFIDENCE LEVEL OF 0.8 OR HIGHER.

	IoU = 0.25					IoU = 0.50				
	Precision	Recall	TP	FP	FN	Precision	Recall	TP	FP	FN
4-camera MoT	0.082	0.589	2332	311	1626	0.855	0.571	2261	382	1697
single-camera MoT	0.887	0.581	2302	292	1656	0.867	0.568	2248	346	1710
Without MoT	0.864	0.330	1307	206	2651	0.845	0.323	1278	235	2680

We focus on the differences between the 4-camera MoT and the single-camera MoT for different IoU values in Table III. As the IoU value decreases, the difficulty of detection decreases and the number of objects judged to be successfully detected increases. When the IoU is changed from 0.5 to 0.25, the single-camera MoT shows a 0.02 increase in Precision and a 0.013 increase in Recall, whereas the 4-camera MoT shows a larger rate of increase, with a precision of 0.027 and recall of 0.018. This is because the 4-camera MoT detects additional objects owing to the lowering of the IoU criteria, resulting in an increase in the number of successful detections. Thus, 4-camera MoT outperforms the single-camera MoT because 4-camera MoT can detect additional objects that could not be detected by the single-camera MoT. These extra objects are vehicles that are close to the vehicle and require high labeling accuracy for fully automated driving. Although the results of the two methods are similar, the 4-camera MoT plays a significant role for the object detection.

We discuss the details of the detection metrics. Figure 6 shows the objects detected by the 4-camera MoT but not by the single-camera MoT. The blue vehicle detected by the green bounding box is an additional object detected by the 4-camera MoT. In the left image, the entire vehicle is visible between the two vehicles in the foreground; therefore, a bounding box that surrounds the entire vehicle can be created. However, in the right image, only part of the vehicle is visible because the blue vehicle is hidden by two cars in the foreground. In this case, because YOLOX creates a bounding box that surrounds the visible area, it creates a bounding box that surrounds only a portion of the vehicle. In contrast, the correct data obtained from Carla creates a bounding box that surrounds the entire vehicle, regardless of whether only a portion of the vehicle is visible. Because of this difference, the performance of the 4-camera MoT is degraded by the definition of the metrics. The blue vehicle in the right image is a well-detected object, but the bounding box surrounds only the visible part of the object because of the YOLOX specification. Even if the results of YOLOX indicate that the detection is successful, a small IoU value causes detection failure.

V. SUMMARY AND FUTURE ISSUES

We proposed and evaluated an improved method for auto annotation for FL in an automated driving environment. Our method uses data from multiple cameras and LiDAR sensors installed in a vehicle to perform the MoT and label objects. YOLOX is used to detect objects and calculate their confidence levels, and then MoT and PointPainting identify identical objects and assign them identical IDs. We updated the labels based on the highest confidence levels for all objects with the same ID, and the labeling accuracy was improved. In future work, we plan to confirm the effectiveness of our proposed method for FL.

Fig. 6. Bounding box for additional objects. left: Entire object is visible, right: Only a part of the object is visible.

REFERENCES

[1] E. Yurtsever, J. Lambert, A. Carballo, K. Takeda, "A Survey of Autonomous Driving: Common Practices and Emerging Technologies," IEEE Access, volume 8, pp. 58443-58469, 2020.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," Proc. of Artificial Intelligence and Statistics, pp. 1273–1282, 2017.

[3] T. Ponn, T. Kroger, F. Diermeyer, "Identification and Explanation of Challenging Conditions for Camera-Based Object Detection of Automated Vehicles," https://www.mdpi.com/1424-8220/20/13/3699 (date 2023-11-21)

[4] S. Dai, S. M. I. Alam, R. Balakrishnan, K. Lee, S. Banerjee, N. Himayat, "Online Federated Learning based Object Detection across Autonomous Vehicles in a Virtual World," Proc. of IEEE Consumer Communications & Networking Conference 2023, pp.919-920, 2023.

[5] C. J. Rapson, B. -C. Seet, M. A. Naeem, J. E. Lee, M. Al-Sarayreh and R. Klette, "Reducing the Pain: A Novel Tool for Efficient Ground-Truth Labelling in Images," Proc. of International Conference on Image and Vision Computing New Zealand 2018, pp.1–9, 2018.

[6] M. Yu, R. Vasudevan, M. Johnson-Roberson, "Occlusion-Aware Risk Assessment for Autonomous Driving in Urban Environments,"IEEE Robotics and Automation Letters, vol. 4, No, 2, pp. 2235–3341, 2019.

[7] Yu Zhao, Yuanbo Shi, Zelong Wang, "The Improved YOLOV5 Algorithm and Its Application in Small Target Detection," Proc. of ICIRA 2022, pp. 679–688, 2022.

[8] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, V. Koltun."CARLA: An Open Urban Driving Simulator," arXiv:1711. 03938, 2017.

[9] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, "You only look once: unified, real-time object detection," Proc. of CVPR'16, pp. 779–788, 2016.

[10] Wiktor Muroń "motpy" github 2022-02-04 https://github.com/wmuron/motpy. (date 2023-07-04)

[11] S. Vora, A. H. Lang, B. Helou, O. Beijbom, "PointPainting: Sequential Fusion for 3D Object Detection," arXiv:1911.10150 [cs.CV]

[12] mmmmaomao "DataGenerator" github 2021-10-28 https://github.com/mmmmaomao/DataGenerator. (date 2023-07-04)

[13] pometa0507 "6th-ai-reference" github 2022-09-01 https://github.com/pometa0507/6th-ai-reference (date 2023-11-21)