Self-supervised classification of dynamic obstacles using the temporal information provided by videos

Sid Ali Hamideche\textsuperscript{1}, Florent Chiaroni\textsuperscript{1,2} and Mohamed-Cherif Rahal\textsuperscript{1}

Abstract—Nowadays, autonomous driving systems can detect, segment, and classify the surrounding obstacles using a monocular camera. However, state-of-the-art methods solving these tasks generally perform a fully supervised learning process and require a large amount of training labeled data. On another note, some self-supervised learning approaches can deal with detection and segmentation of dynamic obstacles using the temporal information available in video sequences. In this work, we propose in addition to classify the detected obstacles depending on their motion pattern. We present a novel self-supervised framework consisting of learning offline clusters from temporal patch sequences and using these clusters as pseudo labels to train a real-time image classifier. The presented model outperforms state-of-the-art unsupervised image classification methods on BDD100K dataset.

I. INTRODUCTION

Building a vehicle with the capacity of driving itself is one of the most interesting and challenging applications of artificial intelligence. An autonomous vehicle must be aware of its environment, like identifying the obstacles, and should be able to make the right decisions and do the appropriate actions. These abilities can be developed independently and then merged to have a total or partial autonomy. Our work falls within the category of environment understanding.

This environment awareness property is acquired through the perception process, which enables interpreting the data provided by different kind of sensors, such as cameras, ultrasonic sensors, and LIDAR. For example the data can be the image of the surrounding obstacles and the perception process can be ensured by an object detection algorithm.

Object detection \cite{1} consists in localizing and classifying objects in images or videos. It is an important feature in autonomous vehicles as it enables for instance to detect obstacles such as pedestrian, cars, or buildings. During the last decade, the deep learning approaches brought the tools for state-of-the-art detection methods. However, the best-performing models in terms of prediction performances are fully supervised, such that they require a large number of annotated data \cite{2} \cite{3} \cite{4}. It turns out that achieving manually the labeling has a cost. In order to reduce this limitation, other types of learning techniques have been investigated, like semi-supervised \cite{5} and weakly-supervised \cite{6} methods, as they can deal with partially unlabeled or noisy labeled data. In general, these techniques do not perform as well as supervised methods, but they are more practical by reducing the need of fully hand-labeled training sets. However, they still need some annotations. A more challenging task is to directly learn on the easily acquirable unlabeled data.

Unsupervised learning is the process of finding unknown patterns in unlabeled datasets. More specifically, clustering methods \cite{7} are unsupervised learning approaches which consist in splitting data into groups containing only similar samples. These clusters can represent the semantics of objects, like cars and persons.

Another technique that does not use manually labeled data is self-supervised learning \cite{8}. Unlike unsupervised methods, this still uses some kind of supervision which could be provided by other sensors or inferred from the data itself. A relevant example could be learning to predict the next frame in videos. This would not be possible if we do not know that frames in videos are temporally ordered.

In this work, we present a new self-supervised method for obstacles classification. The main contribution is:

- Improving state-of-the-art unsupervised image classification methods in the context of autonomous vehicle perception by adding temporal information provided by videos.

To the best of our knowledge, this is the first work that uses the temporal information provided by videos for unsupervised image classification of moving obstacles.

This work consists of two main parts. In the first part, we extract the patches from the videos while keeping their sequentiality. In the second part, we train a model to cluster these sequences and use the clusters as pseudo labels to train an image classifier (Fig. \textsuperscript{1}). Since we prefer to have a variety of objects during short sequences rather than few objects during long sequences, the videos we use are captured by a static camera mounted on a stationary vehicle.

This paper is organized as follows: In the next section \textsuperscript{II} we review the related work. Then, in Sec. \textsuperscript{III} we discuss the motivations of our work and we describe the proposed system. We show some achieved results while comparing to other methods, and we discuss the problems and the difficulties of this task in Sec. \textsuperscript{IV}. Finally we give our conclusion and talk about some perspectives.

II. RELATED WORK

Dynamic obstacles detection: Obstacle detection is an important feature for autonomous vehicles. Different works
have tackled this problem especially for detecting dynamic obstacles using a monocular camera. While state-of-the-art methods are supervised [1], some others instead proposed self-supervised approaches. In [9], Guizilini et al. focused on detecting all the pixels associated with dynamic obstacles. They start by computing the sparse optical flow using SIFT descriptor [10] to match the current frame with the previous one. Then, they use RANSAC algorithm [11] to estimate the camera motion and consequently split the interest points into 2 sets representing respectively static and dynamic obstacles. Finally, they use these 2 sets to incrementally train a Gaussian Process (GP) classifier [12] to extrapolate the previous splitting stage for all the pixels of the image. As an extension to this method, Bewley et al. [13] proposed to separate the detected dynamic obstacles into multiple obstacles. This method adds a clustering phase by applying DBSCAN [14], a density based clustering algorithm, to split the set of the dynamic interest points into multiple sets. Next, it uses a k-NN [15] trained on these sets for the sparsification in order to enable the prediction for all image pixels.

**Images clustering:** Data clustering is an active research topic in machine learning, especially when it comes to choosing the distance function and the appropriate evaluation metrics. Since then, many popular clustering algorithms emerged like K-means [16], GMM [17], or spectral clustering [18]. These methods are relevant for a wide range of problems but it is hard to scale them to a specific one. For example, we cannot expect to have relevant semantic classes by applying K-means directly to the raw pixels of images.

Image clustering is usually combined with visual representations. Some works propose to apply clustering methods to low-level features, like SIFT [10] and HOG [19]. Though, these hand-crafted features often suffer from aspect variations of objects. Other works, inspired by the recent performances of supervised methods and motivated by the feature learning capacity of deep neural networks, use auto-encoder [20] and its variants [21], [22] to learn representations. While these approaches are interesting for numerous problems, it is not the case for complex image classification tasks. It is not guaranteed that the learned features capture the desired semantic meaning.

Recently, self-supervised representation learning appeared to solve pretext task. These pretext tasks are often chosen to be as similar as possible to the original task. An example of such tasks consists of predicting the context [23], [24], colors from gray-scale images [25], or rotations [26]. Some other works exploit the temporal signal, like minimizing the cosine distances between the representations of the same patch from different frames [23], or predicting the motion segmentation from static images [27]. The learned representations are experimentally proved to perform well when used as features extractor with supervised approaches, but they do not provide the solicited semantic classes when used for clustering algorithm such as K-means.

Other works combine the visual representation learning with the clustering in a single model. These methods generally work better as they optimize jointly both objectives. DEC [28] learns simultaneously the cluster centers and the parameters of a deep neural network. To avoid degenerate solution the model performs a pre-training phase. DAC [29] recasts the clustering problem into a binary pairwise-classification where the goal is to predict the same class for similar objects and different classes for dissimilar ones. IIC [30] trains a model to maximize the mutual information between the representations of a paired dataset. A pair can be generated by applying data augmentation processing techniques to the same image. It turns out that IIC has empirically the best accuracy. Besides, IIC [30] uses a paired dataset which enable to associate patches of a given instance from different frames. We decided to apply this method on temporal sequences of images during the proposed unsupervised classification stage.

### III. Method

In this section, we present the proposed approach. We start by introducing the motivation and then we detail its functioning.

#### A. Motivation

We propose to add the temporal information provided by videos to improve the performance of unsupervised classification existing methods as we apply them on detected dynamic obstacles. We do the assumption that visual features, that we can learn on images without using the rest of the potentially associated temporal video sequence, are limited for the following reasons:

- Objects may have different shapes and colors depending on the viewpoint and the posture; e.g., the front-view of a car does not have the same shape as its side-view.
- We can see different parts of a same object as separate objects until we see it moving. Stationarity can be a limitation of our perception of different objects; e.g., a vehicle with a trailer.
- Objects from different semantic classes may have similar features, like shape and texture, but they can be distinguishable by their way of moving; e.g., persons and cyclists are similar in shape while they move differently.

Using video can help the algorithm to learn an abstract representation of each class that is independent from the viewpoint. For instance, an unsupervised algorithm trained on static data can assign the same label to different objects that are similar when observed from the same side (e.g., the front-view of a car and a truck). It can also assign different labels to the same object observed from different sides, such as a car observed from its front-view and its side-view. Moreover, training on sequences can catch the way the object moves, which could help the model to better distinguish between objects from different classes. For example, a person moves differently from a vehicle, as illustrated in Fig.

The proposed self-supervised learning system is composed of 2 consecutive stages: During the former we extract the dynamic objects from the videos. During the later we train...
Fig. 1: Overview of the proposed system outputs. (1) Dynamic obstacles detection, segmentation and tracking from a monocular camera video. (2) Semantic unsupervised classification of dynamic obstacles.

Fig. 2: Informative spatio-temporal transformations of dynamic obstacles. Orange and blue boxes show that a car observed from its front-view may resemble more to a truck from the front-view than the same car from another point of view. This also shows the difference in movement between vehicles and persons which are respectively solid and deformable objects.

a model to predict the class of those extracted objects (Fig. 1).

B. Dynamic obstacles extraction

The goal in this part is to successively detect, track and extract the patches containing the dynamic objects and group them into sequences containing the same instance. We adapt to our problem the approach previously proposed by Bewley et al., 2014 [13]. Consequently, we design our dynamic obstacles extraction module as follows:

a) Sparse optical flow computing: In order to detect the moving obstacles, the first step is to compute the optical flow by matching the interest points of the current and the previous frame using a paralyzed version of SURF [31].

b) Static and dynamic obstacles separation: The goal here is to split the set of interest points, computed in the previous step, into 2 sets, one containing the static points and the other containing the dynamic ones. As we deal with videos obtained by a static camera (absence of camera movement), the partitioning in this step is straightforward. The ideal optical flow must be 0 for static points and must be different from 0 for dynamic points. However, in practice some matching noise can deteriorate the process. Consequently, we propose to use an empirical threshold close to zero to get static points. We also use a second threshold to get the dynamic points and to get rid of the other type of noise. Furthermore, as we work from a static camera recording point of view, we do not need the RANSAC algorithm [11] presented in the original method [13].

c) Dynamic points distinction: In the previous step we acquired the dynamic points of interest that indicate the location of the moving obstacles. However, to be used for classification, we gather them by dynamic obstacles instances. In this part we follow the original method [13], which uses DBSCAN [14], to split the dynamic set into multiple sets corresponding to different obstacles. The main advantage of this method is that it does not require the number of clusters to be defined. The algorithm takes as inputs the set of the dynamic points represented by their position and optical flow velocity.

d) Obstacles tracking: To associate patches of obstacles observable from consecutive frames we use the tracking algorithm Simple Online and Realtime Tracking (SORT) [32]. It is robust in terms of multiple object tracking accuracy and execution time, to the previously computed dynamic points.

e) Obstacles Segmentation: For the segmentation we use a background subtraction method [33] on the whole frame and then mask the result with bounding boxes containing the dynamic obstacles.

f) Patch grouping: Since our main goal during the first-stage is to retrieve the instances of dynamic obstacles through the video, the last step of this part is to use the computed dynamic obstacles from the previous steps to extract all the patches from all images then group them into sets where each set contains the sorted patches of the same tracked instance. We extract square patches corresponding to the computed bounding boxes.

We sum up these steps in Fig. 3. After having extracted moving obstacles, the second stage consists of predicting the corresponding classes.

C. Obstacles classification

In this section we explain classification approach which consists of using unsupervised learning techniques to cluster the sequences of tracked obstacles, and then use the obtained clusters as pseudo labels to train a classifier on images. This enables an immediate prediction using single patches instead of temporal sequences.

1) Unsupervised clustering of sequences: In an unsupervised setting, clustering of real-world images is a challenging
task. In this part we use IIC [30] for the reasons stated in Sec. II. We recall that image clustering methods are sensitive to unbalanced training data. To avoid degenerate solutions like having one cluster containing all obstacles, these methods tend to learn a uniform distribution where each class is represented by the same number of instances. In this part, we explain how we used IIC to adapt it to our case and how the temporal information is added.

In order to train IIC [30] for clustering we use patch sequences extracted in the first part of our approach instead of single patches. As IIC needs a paired dataset, we generate pairs using two patch sub-sequences extracted from the same patch sequence as illustrated in Fig. 4. This gives the model a prior on how the obstacle moves, and enables to associate identical obstacles from different view-points and postures.

As it is trained on sequences, this deep clustering model cannot predict the classes for single patches. Consequently, in the next section we use the clusters as pseudo-labels to train an image classifier.

2) Images classification: As explained previously we use IIC [30] trained on instance sequences to generate pseudo labels. For each instance, that is represented by a sequence of images containing the same instance as shown in the top of Fig. 3, we average over the softmax outputs of all possible sub-sequences \( s_{ij} \). Then, we assign the class with the highest probability to all images contained by the sequence. With these labels, we train an image classification model in a supervised manner. The whole process of this part is shown in Fig. 5.

IV. Experiments

In this section, we start by describing selected datasets and the learning system settings. Then we present some empirical experiments and the corresponding qualitative and quantitative results.

A. Datasets

1) Training data: As we work with videos captured by a static camera on a vehicle, we used a dataset satisfying this specific requirement. BDD100K [34] contains 100,000 40-second videos in 30 frames-per-second which makes a total of 120,000,000 images. The dataset also provides GPS/IMU which we used to extract the frames when the vehicle is stationary. As the task is already difficult, we chosen, for a first step, to use only videos with clear weather and which are recorded in the daytime. The total number of exploited frames is 100109.

2) Test data: The BDD100k dataset [34] also provides bounding box annotations of the frame at the 10th second of each video. In order to evaluate our method, we use these bounding boxes to build a new dataset of images containing only one object similarly to the first part of our method, but here, we instead use the ground truth bounding boxes. We exclusively extracted potentially moving obstacles of the 8 classes (Bicycle, Bus, Car, Motorcycle, Person, Rider, Train, Truck. Examples in Fig. 6). The classes distribution of the new built dataset is shown in Fig. 7.

B. Patches extraction

Because of light changes and other noises impacting the images matching, the dynamic obstacle detection method may produce a lot of false positives, such that some bounding boxes do not contain any dynamic obstacle. However, many of them can be filtered out by only keeping the boxes that are persistent through the frames. Consequently, we decide to keep patches that appear at least in 3 consecutive frames. We also used the SORT [32] tracker to detect and reject the colliding objects. It turns out that sequences with length of 3 frames enable to capture dynamic obstacles movement patterns, while preserving enough data to train our model. The total number of patch sequences extracted is 3607, with an average of 4.84 images per instance. On another note, if we do the assumption that videos follow the same classes distribution than the created test dataset, then the distribution of bounding box images (i.e. patches) automatically extracted is probably considerably unbalanced. For instance, the car class may be more represented than other classes.

C. Inputs of the deep neural network models

To train the first model which is the one used to generate pseudo labels, we use input sub-sequences of 3 images. We then apply the same random transformation to the patches of the hole sub-sequences. We use the data augmentation techniques proposed in the original implementation of IIC [30]. This includes the random cropping and resizing to get fixed size \( 64 \times 64 \). We also apply Sobel filter as proposed in IIC [30] and in DeepCluster [35]. This helps to prevent
Fig. 4: Clustering training using temporal sequences of instances. (1) Randomly sampling two sub sequences from a given whole sequence to generate pairs. (2) IIC model [30] training.

Fig. 5: Image classification training process. (1) Selection of all the possible sub-sequences for a given sequence. (2) Trained IIC model [30] predictions for all sub-sequences, and average estimation of these predictions in order to assign a cluster to all the images contained by the corresponding whole sequence. (3) Creation of the classifier training dataset using the computed pseudo-labels corresponding to the previously identified clusters. (4) Image (i.e. patch) classifier training.

Fig. 6: Some dynamic obstacle patches from BDD100k [34].

The model from clustering based on trivial cues such as color. This encourages instead to use more meaningful cues like shape. The input of the model is $6 \times 64 \times 64$, where 6 represents the Sobel filter applied to approximate the 2 derivatives along the horizontal and vertical pixel axis of the 3 images ($2 \times 3$).

For the classifier, we use the original images as the input. In other words, the input is $3 \times 64 \times 64$ with 3 representing the color channels number. We also apply data augmentation techniques such as random horizontal flip, color jitter, and random cropping.

D. Training

We use the same configuration as IIC [30] to train our sequences clustering model. We use the Adam optimizer [36] with a learning rate $10^{-4}$, and the objective function of both the clustering and the over-clustering is optimized alternatively. For the classifier, we use the basic SGD optimizer with a learning rate $10^{-3}$ and the Cross entropy loss function.

E. Evaluation metrics

In our experiment, similarly to other works on unsupervised images classification, we use the standard clustering accuracy metric. It consists of finding the best one-to-one mapping between the ground truth labels and the clusters.
TABLE I: Evaluation of our method and state-of-the-art methods for multi-class classification. *i.p indicates that the method is applied on individual patches, while *s indicates that it is applied on sequence of patches.

| Method                  | NMI   | ARI   | V-measure | ACC   |
|-------------------------|-------|-------|-----------|-------|
| PCA [37] + K-means [16] *i.p | 0.0656 | 0.0339 | 0.0656    | 0.1449 |
| PCA [37] + GMM [17] *i.p  | 0.0683 | 0.0373 | 0.0683    | 0.142  |
| PCA [37] + K-means [16] *s  | 0.0831 | 0.0457 | 0.0622    | 0.1387 |
| PCA [37] + GMM [17] *s    | 0.0577 | 0.0195 | 0.0786    | 0.1353 |
| IIC Original [30]        | 0.1924 | 0.1063 | 0.1907    | 0.2353 |
| Our method               | 0.2224 | 0.1312 | 0.2216    | 0.3127 |

TABLE II: Balanced accuracy score for the binary classification car-versus-person.

| Method      | ACC   |
|-------------|-------|
| IIC [30]    | 56.95 |
| Ours        | 82.98 |

We also use the adjusted mutual information (AMI), the adjusted round index (ARI), and the V-measure. In order to better evaluate our unbalanced dataset, we apply up-sampling before the evaluation.

F. Results analysis

We compare the proposed approach exploiting the temporal information with the state-of-the-art image clustering method IIC [30]. As the patches automatically extracted with the presented detection algorithm can be different from the ground truth, we assume that the clustering algorithm learns the same classes. We evaluated the prediction performances of our method with two settings. The former is a multi-class classification and the latter is a binary classification between car and person.

1) Multi-class classification: As shown in table I, the proposed method outperforms the original IIC [30] applied on single patches. Fig. 8 shows the proportions of classes over the clusters. We can see that most clusters are represented by the car class. As stated before, this is due to the unbalanced training dataset. However, we can observe that the proposed method predicts more homogeneous clusters than IIC applied on single patches. For example, with our method the class person is mainly concentrated in two clusters while being relatively more dispatched over all the clusters with IIC.

2) Binary classification: Table II shows that the proposed method outperforms the original IIC [30] as well in binary classification. In this task, the difference in performance is more apparent. We believe that this is due to the difference in terms of movement patterns between a car and person which is only observable by temporal information. The histograms in Fig. 9 also show that our method is able to detect the similarities between instances of the class person, as they are mainly concentrated in one cluster. The class car is dispatched due to the training dataset being unbalanced. Fig. 10 shows obstacles patches with the highest probabilities to be in their associated cluster.

G. Semantic segmentation

To sum up on these experiments, Fig. illustrates some output predictions of the proposed complete framework when
Fig. 11: Examples of our method applied to videos for temporally self-supervised moving obstacles instance detection, segmentation, and motion pattern classification.

| Proposed Method                  | Bewley et al. [13] |
|----------------------------------|--------------------|
| Sparse optical flow computing    | 0.03               | 0.35               |
| Static and dynamic obstacles separation | 0.01               | 0.03               |
| Dynamic points distinction       | 0.009              | 0.009              |
| Obstacles tracking               | 0.001              | 2.03               |
| Obstacles Segmentation           | 0.001              |                    |
| Obstacles Classification         | 0.07               |                    |
| **Total**                        | **0.12**           | **2.5**            |

TABLE III: Computational time in seconds of our method compared to Bewley et al., 2014 [13]. The results presented in this table are from our implementation of both methods.

it is applied on videos. It performs detection, segmentation, and unsupervised classification of moving obstacles. We can observe that the cluster assigned to pedestrians bounding boxes is different to the clusters assigned to car vehicles. However, a weakness of the proposed framework is that the detection process does not enable to correctly separate pedestrian instances if they visually overlap with each other. An additional depth map information may help to deal with this issue while improving in the meantime the foreground dynamic obstacles segmentation.

H. Computational cost

The method was implemented with Python 3, OpenCV 4 [31] and PyTorch [38]. We did the experiments on a machine equipped with an Intel Core i-4710HQ CPU, a GTX 970M GPU and 16GB of RAM. In Table III we compare in terms of execution time our method to the method proposed by Bewley et al. [13] as we took inspiration from the latter. We show that we can save 2 seconds by substituting tracking and segmentation steps. This is mainly due to the fact that that we substituted the time consuming k-NN part with a background subtraction algorithm [33] [33] as in our case we work with videos without ego-vehicle camera motion.

V. CONCLUSION AND PERSPECTIVES

To sum up, motivated by the drive to avoid hand labeled training data for the classification of moving obstacles, we have proposed in this article a novel unsupervised image classification approach. It exploits the temporal information concerning the visual pose transformations and motion patterns of the observed moving obstacles. In practice, we have integrated this technique in a self-supervised learning framework in order to jointly detect, segment, and classify the moving obstacles from a monocular camera without any pre-training or hand labeled data. Our empirical study on BDD100K dataset has demonstrated the usefulness and competitiveness of the proposed framework compared to the state-of-the-art techniques in terms of computational cost for the detection and segmentation steps, and in terms of prediction performances concerning the temporally self-supervised image classification task. However, compared to fully supervised techniques using hand labeled data, the proposed model remains limited in terms of prediction performances.

Consequently, it may be interesting to investigate future research for:

- Improving detection and tracking parts in order to provide more consistent patch sequences in input of the proposed temporal clustering approach;
- Using additional information from a depth sensor in order to better separate and analyze moving obstacles;
- Dealing with unbalanced training data for unsupervised classification without class proportion prior knowledge.

REFERENCES

[1] Z. Zhao, P. Zheng, S. Xu, and X. Wu, “Object detection with deep learning: A review,” CoRR, vol. abs/1807.05511, 2018. [Online]. Available: http://arxiv.org/abs/1807.05511

[2] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in 2009 IEEE Conference on Computer Vision and Pattern Recognition, June 2009, pp. 248–255.

[3] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” International Journal of Computer Vision, vol. 111, no. 1, pp. 98–136, Jan. 2015.

[4] T.-Y. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context.” CoRR, vol. abs/1405.0312, 2014.

[5] Y. C A Padmanabha Reddy, V. Pulabuigari, and E. B, “Semi-supervised learning: a brief review,” International Journal of Engineering and Technology, vol. 7, p. 81, 02 2018.

[6] Z.-H. Zhou, “A brief introduction to weakly supervised learning,” National Science Review, vol. 5, no. 1, pp. 44–53, 08 2017. [Online]. Available: https://doi.org/10.1093/nsr/nwv1106

[7] A. Gulhane, P. L. Paikrao, and D. Chaudhuri, “A review of image data clustering techniques,” International Journal of Soft Computing and Engineering, vol. 2, no. 1, pp. 212–215, 2012.
