Learning nuisances to track pedestrians in autonomous vehicles

Adaimi, George
VITA, EPFL

Alahi, Alexandre
VITA, EPFL

Abstract. Autonomous vehicles rely on an accurate perception module. One of the fundamental challenges is to efficiently track pedestrians surrounding a vehicle to anticipate risky situations. Over the past decades, researchers have formulated the tracking problem as a data association one where they proposed various representations aiming for invariance to nuisances such as viewpoint changes, body deformation, object occlusion, and illumination changes. However, these methods still suffer to address abrupt changes since they do not explicitly model the nature of the nuisances.

In this work, we propose to train a classifier that recognizes these nuisances, more specifically rotational body deformation of pedestrians. We aim to detect this deformation and use it to improve the tracking process.

I. Introduction

Tracking, in self-driving cars, is an important feature used to deduce the speed and direction of a pedestrian or car. It helps in accurately forecasting a detected object's future trajectory. While there is an increasing trend towards self-driving cars, previous research in the field of tracking is still limited in its capacity. Most of the previous work on tracking algorithms try to be invariant to the nuisances that effect its performance such as partial occlusion, lighting change, body deformation, and viewing angle change. Being able to identify and account for such problems during tracking might make the algorithm more robust to different real-life situations.

In order to improve such tracking, we first need to identify the type of deformation that occurs and deal with it accordingly. We start by first dealing with a specific type of deformation caused by the rotation of the person being tracked.

II. Related Work

There has been some work done in detecting the different changes that occur in real-world scenarios. Cheng et al [1] proposed a background model re-initialization (BMRI) method based on luminance change. This method proved useful in detecting a sudden luminance change by first finding whether the intensity value of a frame differs by a specific threshold from the previous frame. Then, a luminance histogram is used from the output of the first step to detect entropy change. Liu et al [2] implemented a rotation-invariant object detector. This is done by using a new feature extractor called Sector-ring HOG (SRHOG) and a classifier called Boosted Random Ferns(BRF). By calculating the gradient scale and orientation at every pixel and grouping into cells to obtain the SRHOG descriptor, features that are invariant to rotation are extracted and classified using BRF. BRF is used over other classifiers due its robustness to illumination.

To the best of our knowledge, not much work has been done for detecting rotation changes that can enhance a robot's tracking algorithm. In this paper, we propose an approach to this problem that is based on the GOTURN architecture[3]. For our purpose, we used the GOTURN architecture and trained it to detect rotation changes.

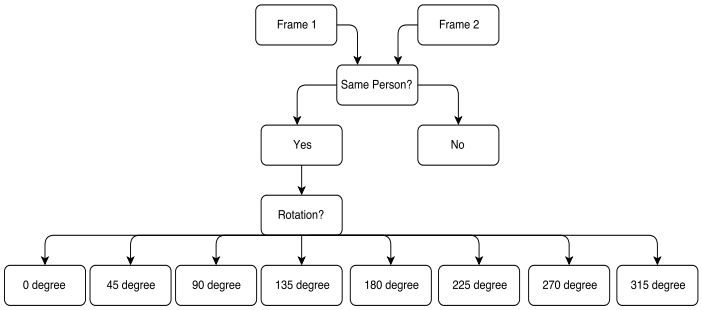

Figure 1: An overview of the implementation

III. Dataset Collection

As can be observed in Figure 1, the final implementation should be able to take in two frames, detect whether it is the same person, and then be able to detect the amount of rotation deformation between these two frames. To achieve this, a labeled dataset with images labeled by their rotation angle is needed. Since no labeled dataset was found, we had to collect our own data and perform some preprocessing steps to create a complete labeled dataset.

Stage 1. Collecting images. Our task requires images of people rotating in front of the camera. Two datasets were found: IAS-Lab RGBD-ID Dataset[4], [5], BIWI RGBD-ID data set[4], [6]. These datasets are RGB-D images of people moving and rotating in a room used for long-term people re-identification.

Both datasets are labeled according to people ID which provides us with one of the labels required by our implementation. The skeletal information of each person in an image is also provided which is used in the next stage to divide the data according to rotation angle.

Stage 2. Filter images by rotation angle. Since the images extracted from the two mentioned datasets are only divided by ID and not by angle, we had to find an automated way to label them by rotation angle. To achieve this, information about the position of the left and right shoulder and the quaternion of the segment joining them to analyze their rotation.

This is first done by setting, for every label, a ground-truth quaternion which is compared to the quaternion of each person in an image. For every image, we calculate the angle difference between its quaternion and that of each label. To be considered part of this label, the difference should not exceed a specific threshold.

This method proved to be somewhat challenging as several problems were encountered. The skeletal information provided does not take into account whether the person is facing the camera or not. This makes it difficult for the algorithm to separate supplementary angles such as 0^o and 180^o. This can be seen in Figure 2.

To overcome this problem, we used a simple face detector implemented by OpenCV[7] which detects the front and side of the face. Thus, when a match for a specific class is found, the image is then passed to the face detector to determine if the predicted class is correct and fix accordingly.

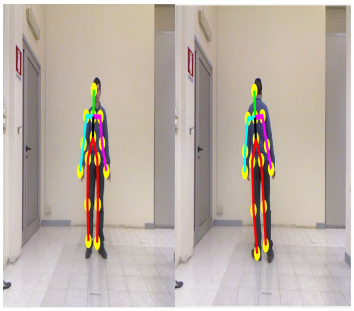

Figure 2: Same alignment of right and left shoulder for two different rotations

Stage 3. Pair and label data. Our use of the dataset requires the data to be paired together before inputing them to our model. There are many ways of pairing the data. The dataset is divided into two sub-datasets, positive and negative, each of which include a different combination of the pairs. The statistics of the different sub-datasets can be found in Table I. An example of each combination is shown in Figure 3 and Figure 4.

Dataset	Number of Pairs
Positive Dataset	482,938
Same IDs + Different/Same angles	482,938
Negative Dataset	266,709
Different IDs + Same angles	33,577
Different IDs + Different Angles	233,132

Table I: The statistics of the positive and negative dataset

(a) Same ID + Same Angle

(b) Same ID + Different Angle

Figure 3: Example from positive dataset

(a) Different ID + Same Angle

(b) Different ID + Different Angle

Figure 4: Example from negative dataset

IV. Network Architecture and Implementation

Our model requires an architecture that uses two images as inputs to compare them together and then outputs a specific label. Convolution neural networks have proven to be the best in analyzing images. Thus, for our initial prototype, we used the same architecture employed by the GOTURN algorithm [3], which in turn is based on the CaffeNet architecture [8], [9], but with a different loss function and final layer.

A. Architecture

As discussed before, the architecture shown in Figure 5 is based on the architecture used in the GOTURN algorithm. Two images are inputed into the network that consists of 5 convolutional layers. The main role of these layers is to extract higher level representations of the image. These representations will provide the important features that are needed to compare the two images. The output of both convolutional layers are concatenated and inputted into a series of three fully connected layers. Their job is to learn the weights that best compare the two images to detect the rotation angle and ID.

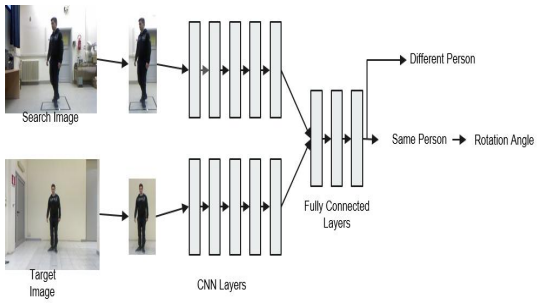

Figure 5: Architecture of CNN used

B. Implementation

Our goal is to implement a model that is able to re-identify a person and regress the angle of rotation of a pedestrian being tracked by a machine.

The architecture is implemented using Tensorflow[10]. Two images are read from the respective dataset and then passed to a pedestrian detector implemented by OpenCV [7]. The detector finds and crops the person in both images which are then inputted to the neural network to be trained. During training, the loss function for detecting rotation computes the softmax

cross entropy between the real label and the predicted label. It follows the equation below:

$$AverageLoss = \frac{1}{|B|} \sum_{X_i \in B} L_i$$

$$where \begin{cases} X_i \in Sample\ Observation \\ L_i = -\ln\left(\sigma(X_i)\right) \\ B \subset Training\ Data \\ \sigma(X_j) = \frac{e^{X_j W}}{\sum_i e^{X_i W}} \qquad \text{(Softmax Equation)} \end{cases}$$

The model aims to minimize this loss during training.

V. Future Results and Conclusion

In this work, we have addressed the problem of detecting rotational deformation encountered in real-life situations, which has not been addressed before. As a first step, we created our own dataset of images of pedestrians and trained a neural network to detect the presence of a rotation in a pair of images. We will then show whether using this information will improve the tracking process. As preliminary testing, results from a neural network that is able to detect rotation angle and another that only detects whether its the same person will be compared. Detecting a rotation angle indicates that the same person was detected. The results will show us the effect of capturing rotation on the performance of tracking a person. We will show results of various loss functions to solve the detections of the different nuisances in the aim of better identifying the person.

In future work, we plan to test our implementation and study its performance in real-life scenario by integrating the algorithm into our lab's robot, Loomo. In addition, there are many other types of nuisances such as viewpoint changes, body deformation , object occlusion, and illumination changes that still need to be considered. Thus, our final goal is to extend our architecture to recognize the different nuisances and improve the tracking process.

References

[1] F.-C. Cheng, B.-H. Chen, and S.-C. Huang, "A background model re-initialization method based on sudden luminance change detection," vol. 38, 11 2014.

[2] B. Liu, H. Wu, W. Su, W. Zhang, and J. Sun, "Rotation-invariant object detection using sector-ring hog and boosted random ferns," *The Visual Computer*, May 2017.

[3] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 FPS with deep regression networks," *CoRR*, vol. abs/1604.01802, 2016.

[4] M. Munaro, A. Basso, A. Fossati, L. V. Gool, and E. Menegatti, "3D Reconstruction of Freely Moving Persons for Re-Identification with a Depth Sensor," in *IEEE International Conference on Robotics and Automation (ICRA2014)*, 2014.

[5] M. Munaro, S. Ghidoni, D. T. Dizmen, and E. Menegatti, "A Feature-based Approach to People Re-Identification using Skeleton Keypoints," in *IEEE International Conference on Robotics and Automation (ICRA2014)*, 2014.

[6] M. Munaro, A. Fossati, A. Basso, E. Menegatti, and L. Van Gool, *One-Shot Person Re-identification with a Consumer Depth Camera*, pp. 161–181. London: Springer London, 2014.

[7] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22Nd ACM International Conference on Multimedia*, MM '14, (New York, NY, USA), pp. 675–678, ACM, 2014.

[9] J. Kuen, K. Lim, and C. Lee, "Self-taught learning of a deep invariant representation for visual tracking via temporal slowness principle," *CoRR*, vol. abs/1604.04144, 2016.

[10] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.