Tracking Methods for Autonomous Driving

Subjects: Computer Science, Software Engineering

Contributor: Florin Leon

Tracking: identifying traffic participants, i.e., cars, pedestrians, and obstacles from sequences of images, sensor data, or observations. It is assumed that some preprocessing of sensor data and/or input images has already been done.

Trajectory prediction: anticipating the future trajectories and motion of other vehicles in order to facilitate navigating through various traffic conditions.

autonomous driving | object tracking | trajectory prediction | deep neural networks | stochastic methods

1. Introduction

Autonomous car technology is already being developed by many companies on different types of vehicles. Complete driverless systems are still at an advanced testing phase, but partially automated systems have been around in the automotive industry for the last few years. Autonomous driving technology has been the focus of multiple research and development efforts by various car manufacturers, universities, and research centers, since the middle 1980s.

A famous competition was the DARPA Urban Challenge in 2007. Other examples include the European Land-Robot Trial, which has been held since 2006, the Intelligent Vehicle Future Challenge, between 2009 and 2013, as well as the Autonomous Vehicle Competition, held between 2009 and 2017. Since the early stages of autonomous driving technology development, research in the related fields has been garnering significant interest in universities and industry worldwide.

2. Tracking Methods

Object tracking is an important part of ensuring accurate and efficient autonomous driving. The identification of objects such as pedestrians, cars, and various obstacles from images and vehicle sensor data is a significant and complex interdisciplinary domain. It involves contributions from computer vision, signal processing, and/or machine learning. Object tracking is an essential part of ensuring safe autonomous driving, since it can aid in obstacle avoidance, motion estimation, the prediction of the intentions of pedestrians and other vehicles, as well as path planning. Most sensor data that have to be processed take the form of point clouds, images, or a combination of the two. Point cloud data may be handled in a multitude of ways, the most common of which is some form of 3D

grid, where a voxel engine is used to traverse the point space. Some situations call for a reconstruction of the environment from the point cloud which involves various means of resampling and filtering. In some instances, stereo visual information is available and disparities must be computed from the left-right images. Stereo matching is not a trivial task and has the drawback that the computations required for reasonable accuracy usually have a significant impact on performance. In other cases, multiple types of sensor data are available, thereby requiring registration, point matching, and image/point cloud fusion. The problem is further complicated by the necessity to account for temporal cues and to estimate motion from time-based frames.

The scenes involved in autonomous driving scenarios rarely feature a single individual target. Most commonly, multiple objects must be identified and tracked concurrently, some of which may be in motion relative to the vehicle and to each other. As such, most approaches in the related literature handle more than one object and are therefore aimed at solving multiple object tracking problems (MOT).

The tracking problem can be summarized as follows: a sequence of sensor data is available from one or multiple vehicle-mounted acquisitions devices. Considering that several observations are identified in all or some of the frames from the sequence, how can the observations from each frame be associated with a set of objects (pedestrians, vehicles, and various obstacles) and how can the trajectories of each such object be reconstructed and predicted as accurately as possible?

Most related methods involve assigning an ID or identifying a response for all objects detected within a frame, and then attempting to match the IDs across subsequent frames. This is often a complex task, considering that the tracked objects may enter and leave the frame at different timestamps. They may also be occluded by the environment or may occlude each other. Additional problems may be caused by defects in the acquired images: noise, sampling or compression artifacts, aliasing, or acquisition errors.

Object tracking for automated driving most commonly has to operate on real-time video. As such, the objective is to correlate tracked objects across multiple video frames, in addition to individual object identification. Accounting for variations in motion comes with an additional set of pitfalls, such as when objects are affected by rotation or scaling transformations, or when the movement speed of the objects is high relative to the frame rate.

In the majority of cases, images are the primary modality for perceiving the scene. As such, a lot of efforts from the related literature are in the direction of 2D MOT. These methods are based on a succession of detection and tracking steps: consecutive detections that are similarly classified are linked together to determine trajectories. A significant challenge comes from the inevitable presence of noise in the acquired images, which may adversely change the features of similar objects across multiple frames. Consequently, the computation of robust features is an important aspect of object detection. Features are representative of a wide array of object properties: color, frequency and distribution, shape, geometry, contours, or correlations within segmented objects. Nowadays, the most popular feature detection methods involve supervised learning. Features start out as groups of random values and are progressively refined using machine learning algorithms. Such approaches require appropriate training data and a careful selection of hyperparameters, often through trial-and-error. However, many results from

the related literature show that supervised classification and regression methods offer the best results both in terms of accuracy and robustness to affine transformations, occlusion, and noise.

3. Trajectory Prediction Methods

In order to navigate through complex traffic scenarios safely and efficiently, autonomous cars should be able to predict the way in which the other traffic participants will behave in the near future with a sufficient degree of accuracy. The prediction of their motion is especially difficult because there are usually multiple interacting agents in a scene. Also, driver behavior is multi-modal, e.g., in different situations, from a certain common past trajectory, several different future trajectories may emerge. An autonomous car must also find a balance between the safety of people involved (its own passengers and other human drivers, or pedestrians) and choosing an efficient speed to reach its destination, without any perturbations to existing traffic. Predicting the future state of its environment is particularly important when the autonomous vehicle should act proactively, e.g., when changing lanes, overtaking other traffic participants and managing intersections [45].

Other difficulties come from the requirement that such a system must be prepared to handle rare, exceptional situations. However, because of the great number of possibilities involved, it should take into account only a reasonable subset of possible future scene evolutions [92] and often, it is important to identify the most probable solution [93].

Reasoning about the intentions of other drivers is a particularly helpful ability. Trajectory prediction can be treated on two different levels of abstraction. On the higher level, one can identify the overall intentions regarding a discrete set of possible actions, e.g., changing a lane or moving left or right in an intersection. On the lower level, one can predict the actual continuous trajectories of the road users [94].

Trajectory prediction needs to be precise but also computationally efficient [95]. The latter requirement can be satisfied by recognizing some constraints that reduce the size of the problem space. For example, the current speed of a vehicle affects its stopping time or the allowed curvature of its future trajectory so as to maintain the stability of the vehicle. Even if each driver has his/her own driving style, it is assumed that traffic rules will be obeyed, at least to some extent, and this will constrain the set of possible future trajectories [93].

(References would be added automatically after the entry is online)