Cooperative Perception for Pairs of Self-Driving Cars

Aaron Miller

CMU-RI-TR-20-41

August 2020

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Maxim Likhachev, *chair*
Kris Kitani
Achal Dave

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

Abstract

Fully autonomous vehicles are expected to share the road with less advanced vehicles for a significant period of time. Furthermore, an increasing number of vehicles on the road are equipped with a variety of low-fidelity sensors which provide some perception and localization data, but not at a high enough quality for full autonomy. In this work, we develop a fused perception system that allows a vehicle with low-fidelity sensors to incorporate high-fidelity observations from a vehicle in front of it, allowing both vehicles to operate with full autonomy. The resulting system generates perception information that is both low-noise in regions covered by high-fidelity sensors and avoids false negatives in areas only observed by low-fidelity sensors, while dealing with latency and dropout of the communication link between the two vehicles. At its core, the system uses a set of Extended Kalman filters which incorporate observations from both vehicles' sensors and extrapolate them using information about the road geometry. Our perception algorithm is evaluated both in simulation and on real vehicles as part of a full cooperative driving system.

Acknowledgments

First, I would like to express my gratitude to my advisor, Prof. Maxim Likhachev, for his guidance and support throughout the research process. I would also like to thank Prof. Kris Kitani and Achal Dave for their time and helpful feedback as members of my committee. Finally, I would like to thank everyone I worked with at Honda and RobotWits for their help with development, integration, and testing, as well as Kyungzun Rim and the other students in the Search-Based Planning Lab.

Contents

1 Introduction **1**
1.1 Contributions 3

2 Related Work **4**
2.1 Cooperative Driving 4
2.2 Distributed Sensor Networks 5
2.3 Single-Robot Multi-Target Perception and Tracking 5

3 Problem Formulation **7**
3.1 Sensor Suites 7
3.2 Mathematical Formulation 7

4 Fused Perception Approach **10**
4.1 Matching Tracks between Vehicles 11
4.1.1 Efficient Solution for Constant $p^{\mathrm{v}}_{\mathrm{FN}}$ 11
4.1.2 Occlusion-based model for $p^{\mathrm{v}}_{\mathrm{FN}}$ 13
4.2 Extrapolation and Filtering 15
4.3 Summary 16

5 Experiments and Results **18**
5.1 Simulation with added noise 18
5.2 Simulation with a real perception system 19
5.3 Real vehicles 21

6 Discussion **25**
6.1 Tradeoffs in Our Approach 25
6.2 Future Work 26
6.2.1 Modeling or Learning Association and False Negative Costs . 26
6.2.2 Prediction 27
6.2.3 Broader System Context 27

7 Conclusions **28**

Bibliography **29**

List of Figures

1.1 An example scenario with an L2 vehicle engaged in cooperative sensing and planning with an L4 vehicle 2

3.1 Sensor layouts of the two real test vehicles 8

4.1 An example bipartite matching problem, showing the measurements from each vehicle as well as the graph and matrix formulations of the resulting problem 14
4.2 An example scenario for the occlusion-based false negative model . . 15
4.3 The structure of the Extended Kalman filter for each tracked vehicle 16

5.1 Simulation running in VTD 19
5.2 RMS and 99th-percentile errors of the fused perception system with simulated noise 20
5.3 Change in perception error with varying amounts of noise in simulation 20
5.4 Simulation setup in Carla 22
5.5 The trajectory of a vehicle tracked by the perception system on the physical test vehicles 23
5.6 An example detection on the real vehicle 24

List of Tables

5.1 MOTA and MOTP results for fused perception in Carla 23

Chapter 1

Introduction

Vehicles are currently being developed with varying levels of driver assistance and autonomy capabilities. There are already cars on the road today that have some ability to sense their surroundings and provide driver assistance but are unable to drive autonomously without constant human supervision. These cars are designated as Level 2 (L2) vehicles. Meanwhile, Level 4 (L4) vehicles are fully autonomous and do not require human supervision in the areas in which they are approved to drive. Such vehicles are currently in development, but are not yet widely available, and will take a long time to replace existing cars on the road even once they are available. For a long period of time, the road will be shared by L4, L2, and lower capability vehicles.

Standards are already in place for vehicle-to-vehicle (V2V) communication, allowing vehicles to communicate wirelessly, albeit at low bandwidth and limited range. Because of this, it is desirable for a limited number of vehicles with high-fidelity sensors (L4 vehicles) to be able to share information from their own perception systems with less capable (L2) vehicles. This allows an L2 vehicle to achieve "Affordable Autonomy through Cooperative Sensing & Planning" (Figure 1.1), where an L2 vehicle is able to operate autonomously for some period of time without expensive sensors by receiving perception information and a suggested trajectory from an L4 vehicle. Fused perception is necessary in order for such a system to be safe; using only perception information from the L4 vehicle, for instance, might miss vehicles behind the L2 vehicle, such as the vehicle in the left side of the top lane in Figure 1.1.

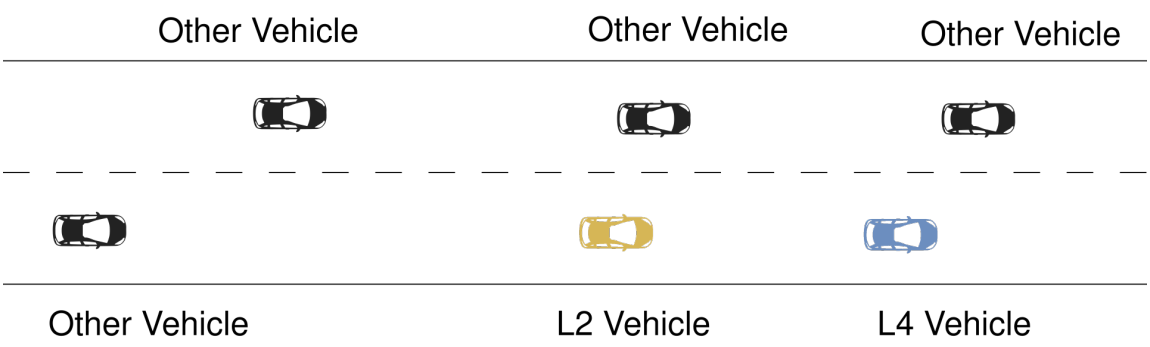


Figure 1.1: An L2 vehicle engaged in cooperative sensing and planning with an L4 vehicle. The L2 vehicle is following the L4 vehicle, and cooperative perception is necessary for the two to have a good state estimate for all the surrounding vehicles.

Failing to detect such a vehicle could cause the system to plan an unsafe lane change for the L2 vehicle. So, the L2 vehicle then fuses the perception information from both vehicles, the L4 vehicle and itself, into a single set of estimates of all surrounding vehicle states. This can then be used to generate a safe plan which follows the L4 vehicle but also avoids obstacles that might not be visible to the L4. This work is concerned specifically with the process of fusing the perception measurements from the two vehicles into a single estimate of the surrounding vehicles' states, and not with other aspects of the system such as cooperative planning or localization.

There are several challenges involved in such a system. First, the bandwidth of a V2V link is not high, meaning that raw sensor information cannot be shared and fused in the same way that sensors would be fused on a single vehicle. Second, there is some latency associated with the network communication. At highway speeds (approximately 55mph and above), this means that measurements of each observed vehicle must be extrapolated separately and accurately for robust results. Third, the system must be able to handle a drop in communication or in quality of perception data from the L4. If another vehicle cuts between the L2 and the L4, or if the L4 goes out of range of the L2, there must be some amount of time where the L2 is able to operate autonomously before the driver is able to take over. And finally, the two vehicles are expected to have different sensor suites with different uncertainties and fields of view. The L4 vehicle in our system has full LiDAR coverage of its surroundings, while the L2 vehicle has more limited and noisier camera and radar coverage.

1.1 Contributions

We present a cooperative perception system which deals with the above challenges. Our system shares observed vehicles ("tracks") and their associated uncertainties along with localization estimates and uncertainties. These estimates are matched to vehicles observed by the L2, allowing elimination of false negatives due to occlusion or limited sensor range. The associated measurements are then fused together in a set of Extended Kalman filters to give high-fidelity estimates of tracked vehicle states at the current time. Because of this EKF architecture, the system degrades gracefully over time if communication drops out, producing reliable perception until the driver is able to take over. Part of our approach is also described in [11]. The system is validated both in simulation and on physical test vehicles equipped with typical L2- and L4-capable sensor suites.

Specifically, our contributions are the following:

- Design of a cooperative perception system for dynamic, highway driving environments
- Development of a model for differing false negatives based on occlusions
- Demonstration and evaluation of the system in simulation and on real vehicles

Chapter 2

Related Work

There has been a good deal of work in both the cooperative driving space and the single-robot perception and tracking space, each of which is laid out in the following sections.

2.1 Cooperative Driving

First, in cooperative driving, one set of approaches has come primarily from a connected vehicles perspective with less emphasis on sensing. Many of these approaches were demonstrated in the Grand Cooperative Driving Challenge [16]. This required competitors to demonstrate various connected vehicle behaviors, such as platooning, using shared localization information from the connected vehicles, but did not require interaction with vehicles only observed by sensors.

Some work has also been done to fuse perception data from multiple vehicles. Several works have done this using occupancy grids [10] [18] [3] or raw point clouds [8]. These approaches successfully combine static obstacles from multiple vehicles, eliminating false negatives and mitigating occlusions. They do have some ability to deal with dynamic obstacles either at low speed or low latency. However, due to the loss of representation of individual vehicles and the lack of velocity information, these approaches cannot fuse observations of other vehicles at highway speeds while tolerating latency in communication.

Other approaches include using Probability Hypothesis Density (PHD) filters

[6], which are able to represent uncertainty both in locations of observed objects and in the number of observed objects. However, these approaches require high communication bandwidth and latency is not considered.

Rauch et al. [17] do fuse individual vehicles by representing them as point clouds, using a single point with uncertainty for each corner of each observed vehicle. This successfully fuses tracks with lower bandwidth requirements, but suffers from higher estimation errors than expected and does not consider latency compensation necessary for operation at highway speeds.

2.2 Distributed Sensor Networks

Work on distributed sensor networks is also relevant to our problem. Several approaches allow for measurements to be made by multiple sensors, or vehicles, connected by a network. These approaches often aim to approximate, or converge to, a global optimum assuming that all measurements are available centrally. Approaches such as the Consensus Kalman filter [13, 14, 15] or the Distributed Kalman filter [25] allow for distributed computation while converging to this global estimate.

The problem can also be approached with more optimization-based methods; for example, [21] derives a distributed optimization-based method that gives better results than the Consensus Kalman filter for distributed tracking in networks of autonomous cars.

The problem of data association in sensor networks has also been studied, and is often used as a component in distributed filtering algorithms. [12] proposes an algorithm which utilizes local matches between pairs of robots and computes global correspondences, resolving inconsistencies caused by incorrect pairwise associations.

2.3 Single-Robot Multi-Target Perception and Tracking

There is a large body of work on data association and filtering for multi-target tracking using sensors on a single vehicle, with either a single sensor or fusing multiple sensors. The data association component of the problem is particularly challenging, even in

the single-vehicle setting. Conceptually, our problem of associating measurements from multiple vehicles is similar to the problem of associating new measurements to existing tracks, which is well studied in the multi-target tracking literature.

There are a variety of methods for defining an objective function for the association problem. A variety of simple pairwise costs can be used, such as those described in [20]. First, there are a variety of possible geometric costs, such as vehicle position, bounding box overlap, or vehicle shape. Also, appearance-based costs can be used, such as distance between visual feature descriptors computed for each detection.

More complex cost functions can also be learned from data. For instance, [19] learns costs based on geometric information from each detection. [26] learns appearance-based costs based on images. [5] uses a more complex association algorithm based on converting the association to a linear program, and also is able to learn the weights for the linear program based on both LiDAR and camera data. Finally, approaches using Graph Neural Networks are able to not only learn costs based on image or LiDAR features for individual detected vehicles, but also model interactions between features computed for different detected vehicles, as demonstrated in [23, 24].

Algorithmically, the optimization is often solved by the Hungarian algorithm [9] [7], which computes the minimum cost matching between two sets, given pairwise costs between objects in the first set and objects in the second set. See for instance [22] for LiDAR-based tracking and [20] for camera-based tracking using Hungarian assignment.

Chapter 3

Problem Formulation

3.1 Sensor Suites

The approach we present is general and can be applied regardless of the types of sensors on each vehicle. With that being said, the domain for the experiments is highway driving. In our experiments on the physical test vehicles, the L4 vehicle is equipped with high-resolution sensors typical of an L4 test vehicle today - LiDAR and Radar covering 360 degrees around the vehicle as well as cameras for perception, and an RTK GPS for localization. The L2 vehicle, on the other hand, is equipped only with Radar and a forward-facing camera for perception and a lower resolution GPS-INS system for localization. Each vehicle is equipped with a DSRC radio for V2V communication. Diagrams showing similar vehicles to the ones that we tested on are shown in Figure 3.1.

3.2 Mathematical Formulation

We start by denoting the L2 vehicle state by $\boldsymbol{x}_t^{\mathrm{L2}}$ and the L4 vehicle state by $\boldsymbol{x}_t^{\mathrm{L4}}$. Similarly, vehicles other than the L2 or the L4 have states $\boldsymbol{x}_t^i$. The state of each vehicle is a vector $\boldsymbol{x} = \begin{bmatrix} x & y & \theta & v & \omega \end{bmatrix}^T$, where x, y, and θ are the pose of the center of the rear axle, v is the signed speed, and ω is the angular velocity. Then, the state of the whole environment can be summarized as $E_t = (\boldsymbol{x}_t^{\mathrm{L2}}, \boldsymbol{x}_t^{\mathrm{L4}}, \boldsymbol{x}_t^1, \boldsymbol{x}_t^2, ..., \boldsymbol{x}_t^N)$.

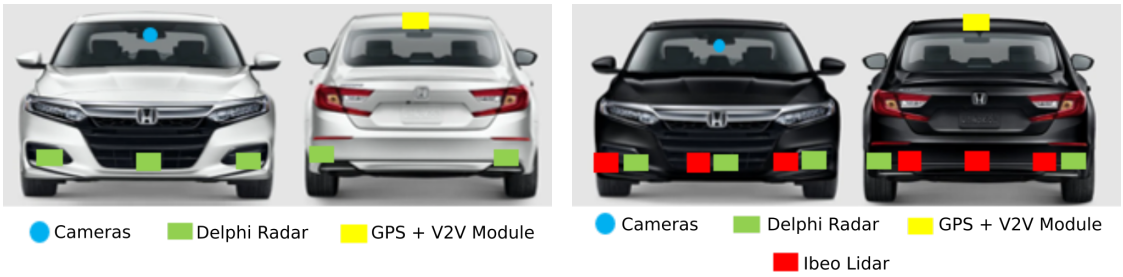


Figure 3.1: Sensor layouts of the two real test vehicles. The L2 is on the left and the L4 is on the right.

Several types of measurements are assumed to be available, and all are assumed to be normally distributed. First, each vehicle has some localization system which provides measurements

$$\begin{aligned} \boldsymbol{z}_t^{\mathrm{L2}} &= \boldsymbol{x}_t^{\mathrm{L2}} + \boldsymbol{\varepsilon}_t^{\mathrm{L2}}, \boldsymbol{\varepsilon}_t^{\mathrm{L2}} \sim \mathcal{N}(0, \Sigma_t^{\mathrm{L2}}) \\ \boldsymbol{z}_t^{\mathrm{L4}} &= \boldsymbol{x}_t^{\mathrm{L4}} + \boldsymbol{\varepsilon}_t^{\mathrm{L4}}, \boldsymbol{\varepsilon}_t^{\mathrm{L4}} \sim \mathcal{N}(0, \Sigma_t^{\mathrm{L4}}) \end{aligned} \tag{3.1}$$

with independent Gaussian noise $\boldsymbol{\varepsilon}_t^{\mathrm{L2}}$ and $\boldsymbol{\varepsilon}_t^{\mathrm{L4}}$, respectively.

We assume that each vehicle has its own perception and tracking system which operates on its own sensors, and the perception measurements we receive are the outputs of these single-vehicle tracking systems. The perception system is assumed to both have some false negatives (i.e. missed detections) and have some error in estimated state of the tracked vehicles. We do not attempt to deal with false positives, i.e. spurious detections; we assume that these can be filtered out by each single-vehicle tracker. False negatives can have a variety of causes, the most common being occlusions and limited sensor range. These false negatives can be eliminated by the combined system, but cannot be expected to be eliminated by either single-vehicle perception and tracking system alone. Formally, we assume there is some probability $p_{\mathrm{FN}}^{\mathrm{L2}}(\boldsymbol{x}_t^i, E_t)$ that track i is undetected by the L2 vehicle's perception system at time t, and similarly probability $p_{\mathrm{FN}}^{\mathrm{L4}}(\boldsymbol{x}_t^i, E_t)$ that track i is undetected by the L4 vehicle at time t, where the subscript FN stands for false negative. For vehicles with states $\boldsymbol{x}_t^i$

which are detected at time t, we have measurements

$$\begin{aligned} \boldsymbol{z}_t^{i,\mathrm{L2}} &= \boldsymbol{x}_t^i + \boldsymbol{\varepsilon}_t^{i,\mathrm{L2}}, \boldsymbol{\varepsilon}_t^{i,\mathrm{L2}} \sim \mathcal{N}(0, \Sigma_t^{i,\mathrm{L2}}) \\ \boldsymbol{z}_t^{i,\mathrm{L4}} &= \boldsymbol{x}_t^i + \boldsymbol{\varepsilon}_t^{i,\mathrm{L4}}, \boldsymbol{\varepsilon}_t^{i,\mathrm{L4}} \sim \mathcal{N}(0, \Sigma_t^{i,\mathrm{L4}}), \end{aligned} \tag{3.2}$$

for vehicles detected by the L2 and L4, respectively, again with independent Gaussian noise $\boldsymbol{\varepsilon}_t^{i,\mathrm{L2}}$ and $\boldsymbol{\varepsilon}_t^{i,\mathrm{L4}}$.

Chapter 4

Fused Perception Approach

The overall algorithm uses a set of Extended Kalman Filters (EKFs), one for each tracked vehicle. The state of each EKF is reinitialized each time a new set of tracks is available from the L4 vehicle; during normal operation, this measurement is delayed by communication latency, and during a drop in communication this measurement is delayed even more. We reinitialize the state instead of fusing the whole stream of measurements from the L4 because the L4 perception system is assumed to have its own filtering scheme, and the measurement we receive is assumed to contain all information from older measurements made by the L4's sensors. More recent measurements from the L2 perception system are then fused in these EKFs to get a current estimate of the tracked vehicles' states. For this to be possible, we must know which track observed by the L2 corresponds to each track observed by the L4, as well as which tracks were only observed by one of the two vehicles. Finding this correspondence is nontrivial, especially when covariances of estimates aren't uniform. The overall perception algorithm (shown in Algorithm 1) then has two essential components: a MATCH function which takes two sets of tracks and returns a set of pairings between them, and a set of EKFs which make up the PREDICT and UPDATE functions used to extrapolate and fuse the individual track states. It uses these to extrapolate the current set of tracks $S_{t-1|t-1}$ to a timestamp t when a measurement is available, producing the set $S_{t|t-1}$, then associate the measurements with existing tracks and update them to produce the set $S_{t|t}$, before repeating the process until the current time is reached.

This algorithm handles drops in communication with no further modifications - when no new messages are being received, it is still able to take the last received measurement from the L4 vehicle and iteratively fuse more recent measurements from the L2; as the time since the communication failure increases, the output gradually becomes noisier and closer to the L2's raw perception output.

4.1 Matching Tracks between Vehicles

Each measurement must be associated correctly with a measurement from the other vehicle; cases where a vehicle is not observed by both the L2 and the L4 should be correctly identified. We first present a method for doing this when the probability of each false negative, p_{FN}^{v}, is assumed to be independent of other vehicles in the scene. In this case, the problem can be solved very efficiently. We also present a more complex model for p_{FN}^{v} which explicitly reasons about occlusions caused by other vehicles.

4.1.1 Efficient Solution for Constant p_{FN}^{v}

In this case, the problem can be formulated as a minimum-weight bipartite matching problem, where a measurement can be matched either to a measurement from the other vehicle or left unmatched. A cost is associated with each of these, coming from the log-likelihood of the measurement under that match.

For a pair of measurements $(\boldsymbol{z}_t^{i,\text{L2}}, \Sigma_t^{i,\text{L2}})$ and $(\boldsymbol{z}_t^{j,\text{L4}}, \Sigma_t^{j,\text{L4}})$, made by the L2 and L4 vehicles respectively, that are matched to each other, the likelihood of the two measurements is

$$\begin{aligned}\mathcal{L} =&[1 - p_{\text{FN}}^{\text{L2}}(\boldsymbol{z}_t^{i,\text{L2}}, E_t)] f(\boldsymbol{z}^{i,\text{L2}}; \hat{\boldsymbol{x}}_t^{ij}, \Sigma_t^{i,\text{L2}}) \\ &[1 - p_{\text{FN}}^{\text{L4}}(\boldsymbol{z}_t^{j,\text{L4}}, E_t)] f(\boldsymbol{z}^{j,\text{L4}}; \hat{\boldsymbol{x}}_t^{ij}, \Sigma_t^{j,\text{L4}}),\end{aligned} \tag{4.1}$$

where $f(\cdot; \mu, \Sigma)$ is the PDF (probability density function) of the multivariate normal distribution. This contains four factors; first is the probability that the L2 perception system successfully detects the vehicle. Next, we have the probability density f. The vector $\hat{\boldsymbol{x}}_t^{ij}$ here is the state obtained by fusing the pair of measurements $\boldsymbol{z}_t^{i,\text{L2}}$ and $\boldsymbol{z}_t^{j,\text{L4}}$; specifically, it is the maximum likelihood state of the vehicle given the two Gaussian measurements $\boldsymbol{z}_t^{i,\text{L2}}$ and $\boldsymbol{z}_t^{j,\text{L4}}$ and their respective covariances. Equivalently,

it can be viewed as the state of an Extended Kalman filter initialized with these two measurements, which is how it will be used in Section 4.2. Then the $f(\boldsymbol{z}_t^{i,\mathrm{L2}}; \hat{\boldsymbol{x}}_t^{ij}, \Sigma_t^{i,\mathrm{L2}})$ factor is the likelihood of the L2 making the measurement of the precise state that it did; it corresponds to a Gaussian PDF centered on the fused vehicle state, with covariance $\Sigma_t^{i,\mathrm{L2}}$, evaluated at the state estimated by the L2 perception system. The second line of Equation 4.1, i.e. the third and fourth factors in the likelihood, are the corresponding values for the L4 perception system's measurement, given the same fused vehicle state $\hat{\boldsymbol{x}}_t^{ij}$.

The cost for pairing tracks i and j together is then the negative log-likelihood

$$\begin{aligned} C(\boldsymbol{z}_t^{i,\mathrm{L2}}, \boldsymbol{z}_t^{j,\mathrm{L4}}) = &- \ln(1 - p_{\mathrm{FN}}^{\mathrm{L2}}(\boldsymbol{z}_t^{i,\mathrm{L2}}, E_t)) \\ &- \ln(1 - p_{\mathrm{FN}}^{\mathrm{L4}}(\boldsymbol{z}_t^{j,\mathrm{L4}}, E_t)) \\ &- \ln f(\boldsymbol{z}_t^{i,\mathrm{L2}}; \hat{\boldsymbol{x}}_t^{ij}, \Sigma_t^{i,\mathrm{L2}}) \\ &- \ln f(\boldsymbol{z}_t^{j,\mathrm{L4}}; \hat{\boldsymbol{x}}_t^{ij}, \Sigma_t^{j,\mathrm{L4}}). \end{aligned} \tag{4.2}$$

Similarly, for a measurement $\boldsymbol{z}_t^{i,\mathrm{L2}}$ from the L2 vehicle that is left unmatched to a measurement from the L4 vehicle, the likelihood is

$$\mathcal{L} = [1 - p_{\mathrm{FN}}^{\mathrm{L2}}(\boldsymbol{z}_t^{i,\mathrm{L2}}, E_t)] f(\boldsymbol{z}_t^{i,\mathrm{L2}}; \boldsymbol{z}^{i,\mathrm{L2}}, \Sigma_t^{i,\mathrm{L2}}) p_{\mathrm{FN}}^{\mathrm{L4}}(\boldsymbol{z}_t^{i,\mathrm{L2}}, E_t). \tag{4.3}$$

The first two factors are identical to those in the paired case, except that the fused position (the center of the Gaussian PDF) is now simply equal to the measured state, because there is only one measurement. The final factor is simply the probability of a false negative from the L4 perception system at that particular state. The cost for assigning track i from the L2 as a false negative from the L4 is then

$$\begin{aligned} C(\boldsymbol{z}^{i,\mathrm{L2}}, \mathrm{FN}) = &- \ln(1 - p_{\mathrm{FN}}^{\mathrm{L2}}(\boldsymbol{z}^{i,\mathrm{L2}}, E_t)) \\ &- \ln f(\boldsymbol{z}_t^{i,\mathrm{L2}}; \boldsymbol{z}_t^{i,\mathrm{L2}}, \Sigma_t^{i,\mathrm{L2}}) \\ &- \ln p_{\mathrm{FN}}^{\mathrm{L4}}(\boldsymbol{z}_t^{i,\mathrm{L2}}, E_t). \end{aligned} \tag{4.4}$$

The cost for assigning a track j measured by the L4 perception system as unmatched to any track from the L2 can be computed analogously.

It should be noted that E_t in these equations is unknown. If $p_{\mathrm{FN}}^{\mathrm{L2}}$ and $p_{\mathrm{FN}}^{\mathrm{L4}}$ are

independent of which other tracks in the problem are matched to each other, then the problem can be modeled as a bipartite graph, as shown in Figure 4.1. The problem is then solved by the Hungarian algorithm [9] [7] [2], which finds the best cost matching in cubic time in the total number of measurements from both vehicles. In particular, we evaluate simply letting $p_{\text{FN}}^{\text{L2}}$ and $p_{\text{FN}}^{\text{L4}}$ be constants. An example matching problem, and its formulation as a bipartite graph to be solved by the Hungarian algorithm, is shown in Figure 4.1.

4.1.2 Occlusion-based model for p_{FN}^{v}

We also develop a more complex model for p_{FN}^{v} that accounts for occlusions due to other vehicles in the scene, as well as for the distance between the sensor and the detected vehicle. Both of these are accounted for naturally by considering the angle ϕ occupied by a vehicle as seen by the sensor; an example can be seen in Figure 4.2. For a lone vehicle with no others around to occlude it, this can be calculated simply from the predicted bounding box of the detected vehicle. If the vehicle is occluded by one of the other observed vehicles in the scene, however, we only count ϕ as the portion of the occluded vehicle which is visible to the sensor.

Then, given ϕ, we assume that $\ln p_{\text{FN}}^{\text{v}} = -\alpha\phi$ for some positive constant α. This is a reasonable assumption; for example, consider a LiDAR sensor with a fixed angular resolution. If detection involves pointwise classification or segmentation, then a false negative may require misclassifying all k points belonging to an object. If we assume that there is a fixed probability p of misclassifying each point, and that these misclassifications are independent, then we have

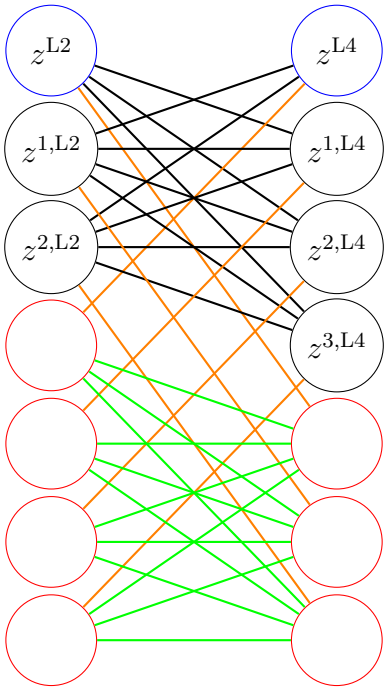
$$p_{\text{FN}} = p^k \implies \ln p_{\text{FN}} = k \ln p.$$

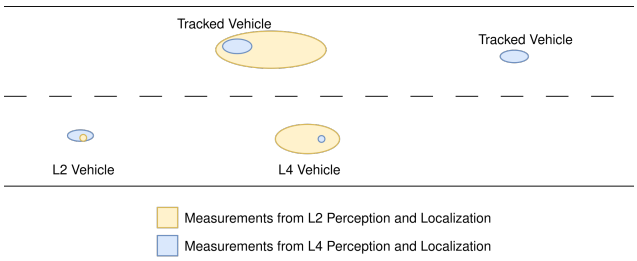
And if the sensor has ρ points per radian, then we have

$$\ln p_{\text{FN}} = k \ln p = \rho\phi \ln p = -\alpha\phi,$$

where $\alpha = -\rho \ln p$ is a positive constant. In practice, we also bound p_{FN}^{v} away from 1 by some amount, so that a probability of 0 is never assigned to a detection.

In the previous section, we were able to solve the problem efficiently because





	z^{L4}	$z^{1,\mathrm{L4}}$	$z^{2,\mathrm{L4}}$	$z^{3,\mathrm{L4}}$	FN	FN	FN
z^{L2}	∞	$\mathbf{-12}$	19964	2154	7	∞	∞
$z^{1,\mathrm{L2}}$	$\mathbf{-12}$	3595	1647	1892	∞	7	∞
$z^{2,\mathrm{L2}}$	4820	2890	1325	$\mathbf{-10}$	∞	∞	7
FN	7	∞	∞	∞	**0**	0	0
FN	∞	7	∞	∞	0	**0**	0
FN	∞	∞	**7**	∞	0	0	0
FN	∞	∞	∞	7	0	0	**0**

Figure 4.1: Example toy matching problem. The scenario is shown in the top right; tracks and localization from the L4 vehicle (with their covariances in x and y) are shown in blue, and tracks and localization from the L2 are shown in yellow. In the representation of the matching problem as a bipartite graph (on the left), each blue node represents a localization measurement, black nodes represent perception measurements, and red nodes represent false negatives. The left column shows the L2 localization and one track detected by the L2, while the right column shows the L4 localization and two tracks detected by the L4 perception system. Each edge represents an allowed connection - no edge between two nodes is the same as infinite cost. Black edges are associated with pairs of measurements and have costs given by the function C. Orange edges represent assigning a detection as undetected by the other vehicle and have costs $-\ln p^{\mathrm{v}}_{\mathrm{FN}}$. Green edges allow FN nodes to be matched with each other for 0 cost, which is necessary for this to be a well-defined bipartite matching problem. The table of edge costs is shown in the bottom right; edges forming the optimal matching are in bold.

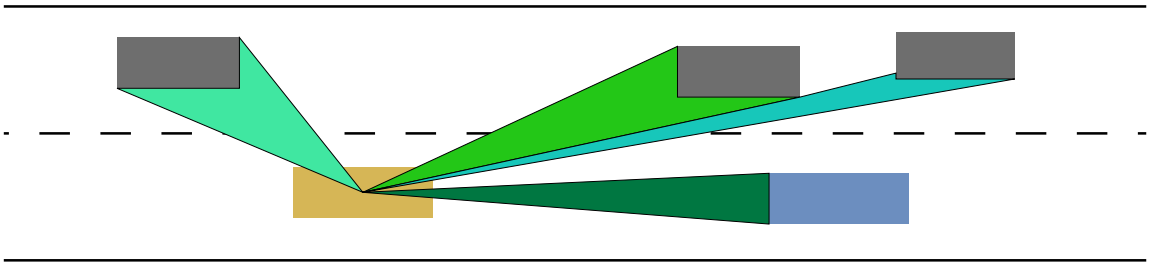


Figure 4.2: An example scenario for the occlusion-based false negative model. The L2 vehicle is shown in yellow on the bottom left, the L4 is in blue on the bottom right, and three other vehicles are in grey. Here we show the visibility of each vehicle to the L2's sensors; a similar diagram could be made for the L4's sensors. The model considers the angle ϕ occupied by each vehicle in the sensor's field of view, based on the predicted bounding box of each vehicle. For occluded objects, only the visible portion contributes to ϕ.

the log likelihood decomposed into a sum over terms which only depended on single vehicles or pairs of vehicles. However, in this case, the angle ϕ that each vehicle in the scene presents to the sensor may depend on arbitrarily many vehicles in the scene. Therefore, the problem can no longer be solved efficiently with the Hungarian algorithm. In our experiments, it is still solvable for small numbers of vehicles by enumeration of possible solutions with simple pruning rules.

4.2 Extrapolation and Filtering

Each tracked vehicle has its own EKF with state space $[x, y, \theta, v, \omega]^T$, where x, y, and θ are the pose in SE(2), v is the signed speed, and ω is the angular velocity. The angular velocity is not expected to be measured; instead, we use the road curvature and vehicle speed v to calculate a prior for ω. The PREDICT function assumes constant speed and angular velocity to extrapolate the given set of states to the desired time, while propagating uncertainty and adding process noise. The UPDATE function is then the standard EKF update which takes the filter state and covariance $(\hat{\boldsymbol{x}}, \hat{\Sigma})$ and measurement $(\boldsymbol{z}, \Sigma)$ and returns the new state and covariance. Because each single-vehicle perception system is assumed to have some internal filtering scheme that would cause sequential measurements to be highly correlated, we do not fuse every measurement we get from the perception system; instead, we subsample down

to a frequency at which sequential measurements are not highly correlated. A diagram of the filter structure is shown in Figure 4.3.

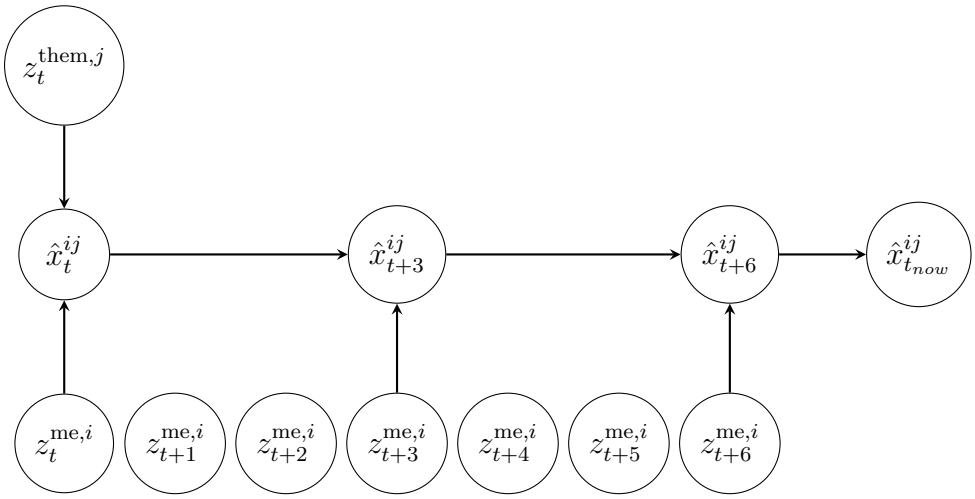


Figure 4.3: The structure of the Extended Kalman filter for each tracked vehicle. This diagram shows the flow of information through the filter corresponding to our measurement of vehicle i, which has been matched with the other perception system's track j. The top row represents the most recent measurement of vehicle j from the tracking system on the other vehicle; only the most recent measurement is used because it comes from a tracking system which is assumed to incorporate all information from earlier times. The bottom row contains all measurements of vehicle i from the tracking system on our own vehicle. The middle row then contains the fused estimates of the vehicle state. First, the two measurements at time t are fused into the state. Next, the state is extrapolated forward by enough timesteps to avoid too much autocorrelation in the tracker output, and another measurement from the tracker on our vehicle is fused. This is repeated up to the most recent measurement, after which the estimate is simply extrapolated up to the current time.

4.3 Summary

The overall fusion process (on the L2 vehicle) is summarized in Algorithm 1. Computation begins each time a new set of tracks is received from the L4 vehicle, at which point the corresponding measurement from the L2 vehicle is retrieved from memory. These two sets of tracks are then matched together, using either the approach in Section 4.1.1 or Section 4.1.2.

Algorithm 1 Fuse Perception on L2

1: t_{L4} = Time of last available measurement from the L4
2: T = Time of last available measurement from the L2
3: M = MATCH($\{\boldsymbol{z}_{t_{\text{L4}}}^{i,\text{L4}}\}$, $\{\boldsymbol{z}_{t_{\text{L4}}}^{j,\text{L2}}\}$)
4: $S_{t_{\text{L4}}|t_{\text{L4}}} = \emptyset$
5: **for** match in M **do**
6: **if** match is a false negative from one vehicle **then**
7: Add one measurement ($\boldsymbol{z}_{t_{\text{L4}}}^{i,\text{L4}}$ or $\boldsymbol{z}_{t_{\text{L4}}}^{j,\text{L2}}$) to $S_{t_{\text{L4}}|t_{\text{L4}}}$
8: **else**
9: Add FUSE($\boldsymbol{z}_{t_{\text{L4}}}^{i,\text{L4}}, \boldsymbol{z}_{t_{\text{L4}}}^{j,\text{L2}}$) to $S_{t_{\text{L4}}|t_{\text{L4}}}$
10: **end if**
11: **end for**
12: **for** $t = t_{\text{L4}} + k, t_{\text{L4}} + 2k, ..., T$ **do**
13: $S_{t|t-1}$ = PREDICT($S_{t-1|t-1}$, t)
14: $S_{t|t}$ = UPDATE($S_{t|t-1}$, $\{\boldsymbol{z}_t^{j,\text{L2}}\}$)
15: **end for**
16: **return** PREDICT($S_{T|T}$, t_{now})

We maintain a set $S_{t_1|t_2}$ of tracks estimated at time t_1 given measurements up to time t_2; this is initialized as $S_{t_{\text{L4}}|t_{\text{L4}}}$ by iterating over the pairs returned by the MATCH function. For vehicles which were detected by both the L2 and the L4, their states are initialized by the result of the FUSE function, which computes the posterior given the two Gaussian measurements. For vehicles which were only detected by one of the L2 and the L4, the single measured state is simply added to the set.

Finally, we step forward in time, fusing every kth set of tracks into the maintained set of vehicle states. Each vehicle state measurement is fused using the EKF PREDICT and UPDATE functions into the correct filter, as determined by the original MATCH results from line 3. The end result is then extrapolated to the current time and returned.

Chapter 5

Experiments and Results

We tested our perception system both in simulation and on real vehicles. Testing in simulation allowed us to study how the system responds to various amounts of latency and to gather statistics on how the system performs. On the real vehicle, we are able to gather some statistics as well. However, we rarely have ground truth positions for the tracked vehicles, so we are primarily concerned with demonstrating that the perception system generates smooth output which eliminates false negatives. In the following subsections, we present results from testing perception in simulation and on real vehicles.

5.1 Simulation with added noise

The first set of experiments were done in a highway driving scenario using the VTD simulator (shown in Figure 5.1). The road is a divided highway with a speed limit of 60mph (26.8m/s) and moderate to heavy traffic traveling at approximately that speed. The simulator generates perception from both the L2 and L4 vehicles by applying limited sensor range to each (200m for the L4 and 100m for the L2), along with Gaussian noise on the poses and speeds of all observed vehicles (.12m and .5m/s on position and velocity respectively on the L4, and .25m and .5m/s on the L2). Similarly, the simulator adds Gaussian noise to the localization available to the system for each vehicle (.01m on the L4 and .1m on the L2). Random latency is also injected into the communication between the vehicles; the typical delay is around 100ms, but

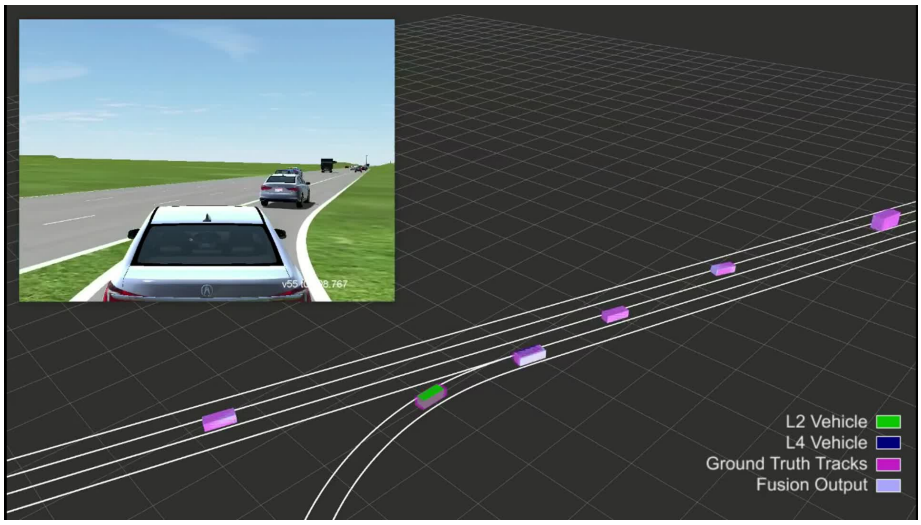


Figure 5.1: Simulation running in VTD in the top left, with the output of the fused perception system shown in the main figure.

there are random spikes of higher latency as well.

First, the RMS error and the 99th percentile error of the fused perception output is shown in Figure 5.2. It remains in a high-quality range (on the level of the L4 perception system) for all typical latencies; for longer delays from the last measurement, as caused by drops in communication, the error increases, but remains in a usable range for long enough that the system can continue operating until the system is able to ask the driver to take control.

The rate of mismatched tracks is also low; in these experiments in simulation, only 0.017% of measurements were incorrectly associated between the L2 and L4 perception systems.

5.2 Simulation with a real perception system

We also do experiments in the Carla [4] simulator, which allows us to simulate multiple vehicles with LiDAR and camera sensors. An example of the system running in simulation is shown in Figure 5.4. We can then run standard single-vehicle perception and tracking algorithms on these sensors to generate more realistic inputs to our system. This is intended to include more realistic failure modes due to occlusions from other vehicles or seemingly random false negatives. Because of this, we are

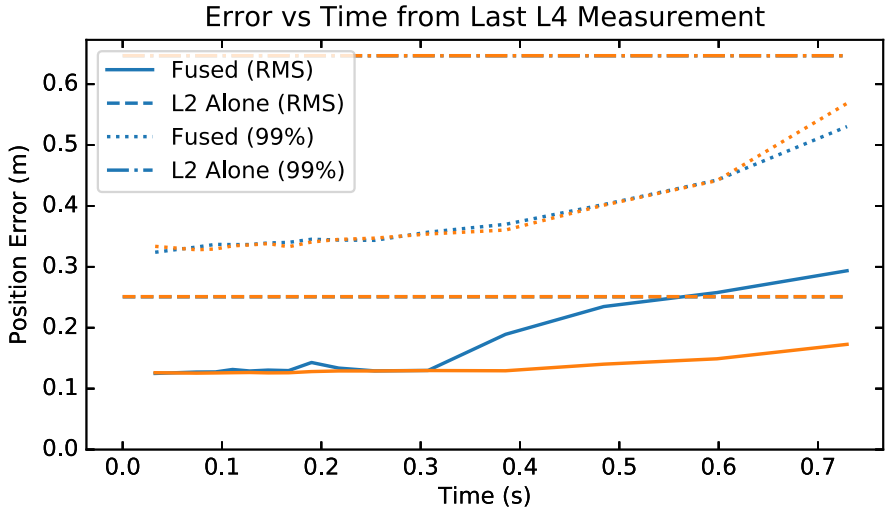


Figure 5.2: RMS and 99th-percentile errors between the ground truth track locations and output of the fused perception system in VTD with simulated noise. Position error is split into on-track (blue) and cross-track (orange) error. The horizontal axis shows the time since the last measurement from the L4 perception system; the average error increases with time, but remains near or below the L2 perception system on its own up to 1s of latency.

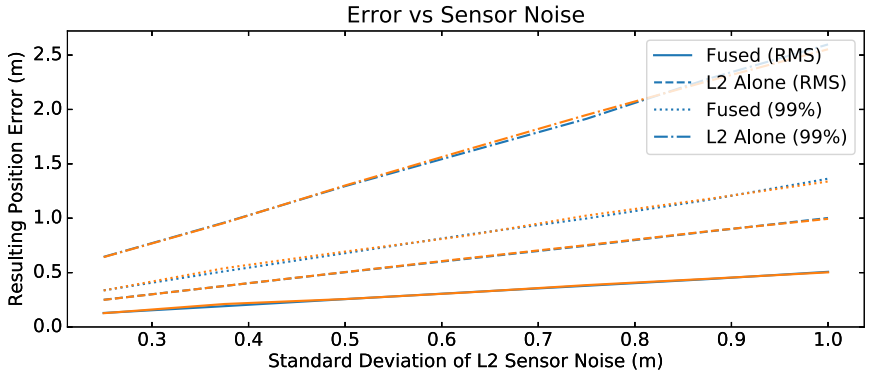


Figure 5.3: Change in perception error with varying amounts of noise in simulation, with L4 and L2 sensor noise increased in proportion to each other. As in Figure 5.2, blue and orange are on-track and cross-track error respectively. Error increases linearly with sensor noise, as would be expected.

able to compare the fused perception system with and without the occlusion-based false negative model. Here, we evaluate using metrics inspired by MOTA and MOTP [1] from the multi-target tracking literature. The standard MOTA metric penalizes false negatives, false positives, and identity switches; our system is matching between detections from two vehicles instead of across time, so we do not measure identity switches, and only include false negatives and false positives. It should be noted that while we assume the underlying single-vehicle detection and tracking systems to not generate a significant number of false positives, the fusion system can and does generate false positives; this occurs whenever two detections of the same physical vehicle are not matched and are output from the fusion system as two separate tracks. MOTP is calculated as usual, with the distance being the 2D distance in the ground plane.

The results are shown in Table 5.1. MOTA is low for all the systems tested, because the set of ground truth vehicles in the simulator includes vehicles not visible to sensors on either the L2 or L4. Because of this, the interesting point is the comparison between the single-vehicle MOTA and the MOTA of the output of the fusion system. As shown in the table, the MOTA of the fusion system is significantly higher than the MOTA of either the L2 or L4 vehicles alone, indicating that our system is correctly matching the tracks observed by the L2 and L4 vehicles.

There is not a significant difference in performance between the occlusion-based and constant models for p_{FN}^{v}; this is partially because the detection algorithms tested were not very high-quality, giving seemingly random false negatives not caused by occlusions. Further experiments using either higher quality detectors or more sophisticated false negative models could give more conclusive results; this is discussed further in Section 6.2.1.

5.3 Real vehicles

Experiments were run on data from real vehicles on a test track and on a public road. The test track is similar to a divided highway with three lanes. Vehicles were not driving at highway speed, but instead at approximately 12m/s on the test track and up to 18m/s on the public road.

The perception system was tested on several scenarios on the test track and a

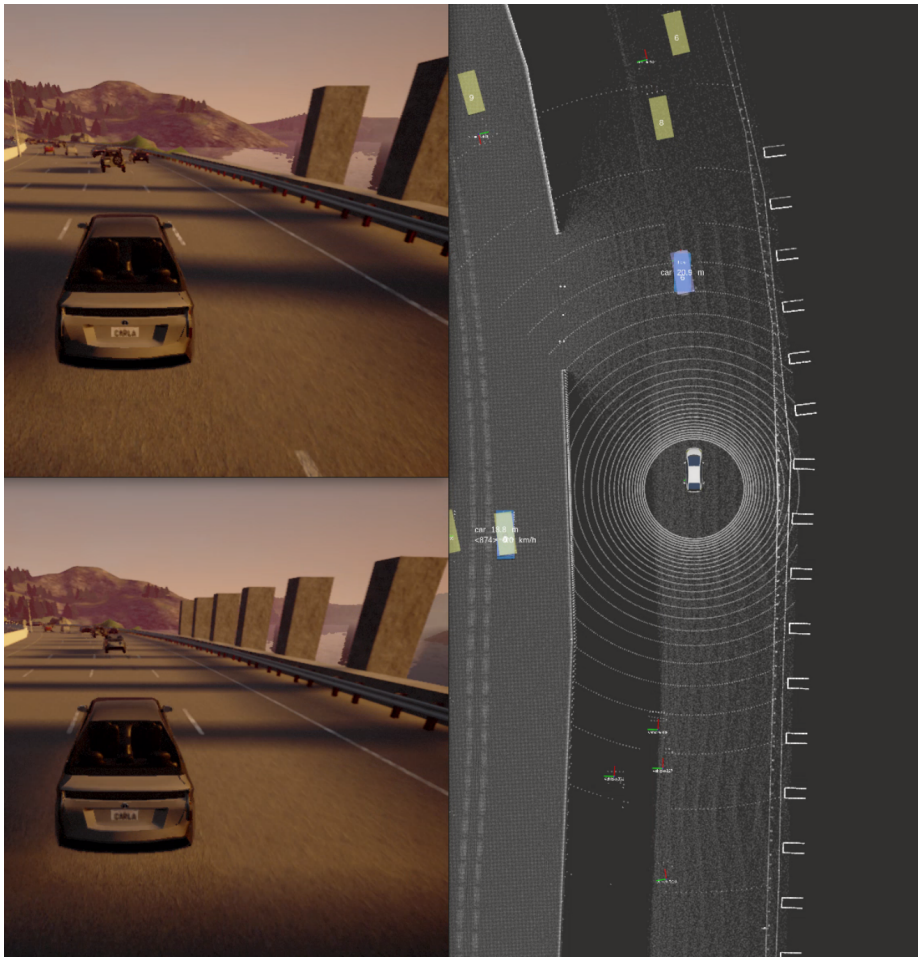


Figure 5.4: Cooperative perception simulation in Carla. On the top and bottom left, third-person views of the lead L4 vehicle and the following L2 vehicle are shown, respectively. The right side shows the corresponding scenario in RViz. The L2 vehicle is at the center of the screen, with the L4 vehicle above. Tracks from the L2 vehicle perception system are in blue and tracks from the L4 perception system are in yellow. Other vehicles can be seen by the coordinate axes at their ground truth locations, for example at the bottom of the figure.

	L2 Alone	L4 Alone	Occlusion-based p_{FN}^{v}	Constant p_{FN}^{v}
MOTA	0.106	0.129	0.201	0.208
MOTP	0.519	0.507	0.491	0.512

Table 5.1: MOTA and MOTP results for fused perception in Carla. The method of evalution is described further in section 5.2. In terms of MOTA, the fused perception system (in the two rightmost columns) achieves much better results than either the L2 or L4 perception systems alone, indicating that the system successfully eliminates false negatives as intended. In terms of MOTP, the fused perception system is approximately equivalent to each indiviual perception system.

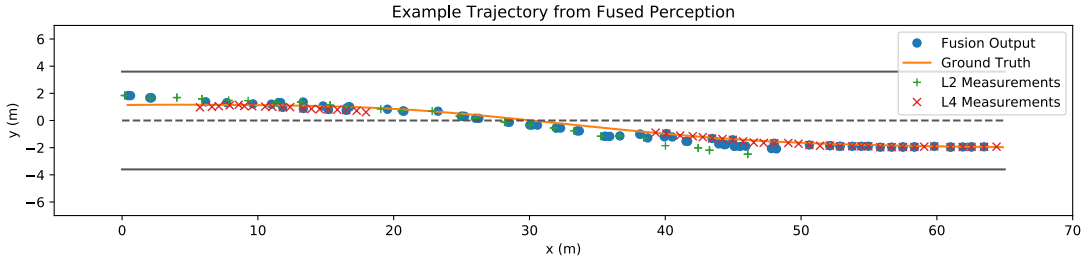


Figure 5.5: The trajectory of a vehicle tracked by the perception system. The vehicle is traveling from left to right while changing lanes. Both the raw input from the L2 and L4 systems are shown, as well as the output of the fusion system. The fusion system successfully deals with false negatives from both perception systems, while also having less error than the L2 system alone. Because the tracked vehicle also has an RTK, we use this to show the ground truth trajectory as well.

public road, with other vehicles in each scenario. The perception system successfully matched the vehicles from the L2 and L4 perception systems correctly, eliminating false negatives (which were due to occlusions and sometimes to blind spots). An example detection is shown in Figure 5.6. Because we did not have access to ground truth poses or velocities for most of the tracked vehicles, we do not study the accuracy of the predicted tracks on the real data. However, we do have one example of the system tracking a vehicle equipped with an RTK to provide ground truth position; this is shown in Figure 5.5, where the system handles false negatives from both the L2 and L4 perception systems at different times and generates a smooth output trajectory close to the ground truth.

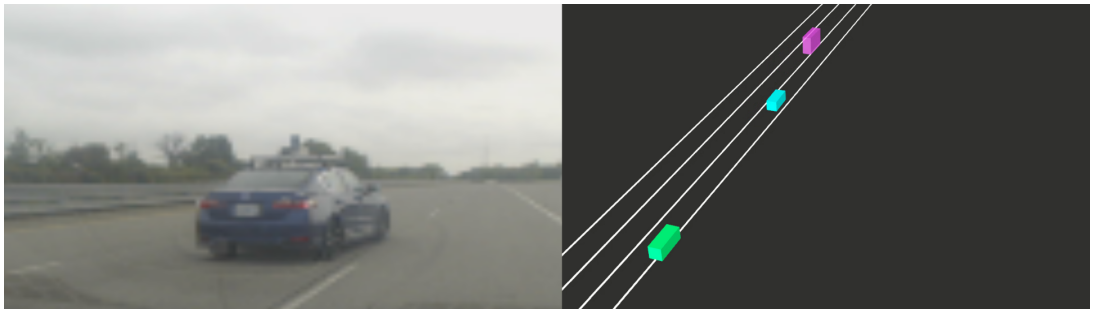


Figure 5.6: An example detection on the real vehicle. The image on the left is from a camera in the L4 vehicle (not the camera used for detection). On the right, the detected vehicle is seen in pink, the L4 is shown in blue, and the L2 is in green.

Chapter 6

Discussion

On average, our system performs well; however, there are a variety of tradeoffs present, and potential for future work to improve on our approach in multiple ways.

6.1 Tradeoffs in Our Approach

First, we use only the measured vehicle states (pose and velocity) in the matching process. This has the advantage of requiring minimal network bandwidth between the L2 and L4 vehicles, which is useful because typical DSRC radios such as the one on our physical test vehicles do not have much bandwidth available. It also has the advantage of being fairly independent of the specific sensor or set of sensors making the measurement, up to differing sensor uncertainties. However, using only vehicle pose and velocity can sometimes result in ambiguous matching problems, especially with the L2 vehicle having lower quality sensors. For this reason, single-robot MOT approaches often use additional features, as discussed in Section 2.3.

Second, there are no components in our system that are learned from data. There are some advantages to this approach; the output of our approach is very explainable, while methods such as neural networks can make predictions that are difficult or impossible to explain or diagnose. For autonomous driving applications where safety is a concern, explainability can be an important factor. Furthermore, approaches that are learned from data by definition require a dataset, and can behave unpredictably in situtations which are not present in the training data. There are also advantages

to using learned components in such a system; learned components can be more expressive and capture effects that are very difficult to model explicitly. Possible methods of integrating learned components are described in the following section.

6.2 Future Work

6.2.1 Modeling or Learning Association and False Negative Costs

Both the pairwise cost of associating two tracks and the cost associated with a false negative can be improved by future work. First, future work could investigate the use of non-Gaussian distributions to model the errors in perception more accurately. However, even arbitrary distributions over measured state vectors (pose and velocity) may not be expressive enough. There are possible additions to the matching cost, such as other cost terms based on geometric or appearance information used in the multi-target tracking literature, which may help with the matching process.

In addition, learning either the pairwise cost or the false negative cost from data could offer better performance. For the false negative cost in particular, the failure modes of a given detection and tracking pipeline are difficult to model explicitly, but may be easier to learn with a function approximator such as a neural network. A system which learns both of these pairwise and unary costs would still have to respect the constraints of the problem setup: first, low communication bandwidth; and second, differing sensor suites on the L2 and L4 vehicles. So, a viable approach may be to learn separate encoders for the L2 and L4 vehicles which map relevant information about each detection (such as appearance information from the available sensors) into a low-dimensional feature space. This would allow these low-dimensional features to be shared across the network, instead of raw sensor data. Furthermore, recent work [23, 24] on the single-vehicle multi-target tracking setup has shown that Graph Neural Networks (GNNs) can also be used on top of feature extractors to learn interactions between multiple vehicles in the scene and produce better cost estimates for association. An approach such as this could be trained end-to-end, in our case training the two different feature extractors for the L2 and L4 vehicles, as well as a GNN on top of those feature extractors.

6.2.2 Prediction

Another potential area for improvement is in the prediction model we use in the Extended Kalman filters, as described in Section 4.2. So far, we have used a constant-velocity model with Gaussian process noise; this has the advantage of allowing us to easily maintain a unimodal, Gaussian representation of the predicted vehicle state in an EKF. However, vehicle behavior is often much more predictable, albeit often multimodal. Recent work in prediction and forecasting for autonomous driving has made progress on much more sophisticated prediction models, often using not only the current vehicle state, but also information about the road geometry and connectivity as well as other vehicles in the area. An extension to such prediction models would also require using a more sophisticated filter to represent a likely multimodal vehicle state distribution, such as a particle filter. Use of more sophisticated prediction models could allow the system to tolerate higher latency, or maintain good estimates for longer after a loss of communication.

6.2.3 Broader System Context

Our work has considered the problem of fusing perception data while an L2 and L4 vehicle are connected, and for a short period of time after a loss of communication. However, this is only one component of several required to deploy such a system in the real world. Work has also been done on cooperative localization (described in [11]) and cooperative planning for this system. However, the problem of determining whether it is possible for a particular L2 vehicle to pair with a particular L4 vehicle on the road would still need to be explored, as well as the problem of initializing and terminating cooperative driving, potentially with multiple eligible vehicles on the road at a time.

Chapter 7

Conclusions

We presented the cooperative perception problem as a necessary component of a full cooperative driving system, aimed at allowing an L2 vehicle to operate autonomously by pairing with and following an L4 vehicle, with both vehicles engaged in cooperative perception, localization, and planning. In particular, we focused on highway driving scenarios, where the two vehicles could travel in the same direction for a longer period of time. We developed approaches for cooperative perception in these scenarios, capable of combining and filtering measurements from both the L2 and L4 vehicle, while dealing with latency and drops in communication. Our method eliminates false negatives from either the L2 or the L4, producing a single set of tracks covering the combined area visible to the two vehicles. This approach was demonstrated successfully both in simulation and on data from real vehicles as part of a full cooperative driving system.

Bibliography

[1] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008. 5.2

[2] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, September 2016. doi: 10.1109/ICIP.2016.7533003. 4.1.1

[3] Federico Camarda, Franck Davoine, and Veronique Cherfaoui. Fusion of evidential occupancy grids for cooperative perception. In *2018 13th Annual Conference on System of Systems Engineering (SoSE)*, pages 284–290, June 2018. doi: 10.1109/SYSOSE.2018.8428723. 2.1

[4] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017. 5.2

[5] Davi Frossard and Raquel Urtasun. End-to-end learning of multi-sensor 3d tracking by detection. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 635–642, 2018. 2.3

[6] Jonathan Gan, Milos Vasic, and Alcherio Martinoli. Cooperative multiple dynamic object tracking on moving vehicles based on Sequential Monte Carlo Probability Hypothesis Density filter. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2163–2170, November 2016. doi: 10.1109/ITSC.2016.7795906. 2.1

[7] Roy Jonker and Ton Volgenant. Improving the Hungarian assignment algorithm. *Operations Research Letters*, 5(4):171–175, October 1986. ISSN 0167-6377. doi: 10.1016/0167-6377(86)90073-8. 2.3, 4.1.1

[8] Seong-Woo Kim, Baoxing Qin, Zhuang J. Chong, Xiaotong Shen, Wei Liu, Marcelo H. Ang, Emilio Frazzoli, and Daniela Rus. Multivehicle Cooperative Driving Using Cooperative Perception: Design and Experimental Validation. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):663–680, April

2015. ISSN 1524-9050. doi: 10.1109/TITS.2014.2337316. 2.1

[9] Harold W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. ISSN 1931-9193. doi: 10.1002/nav.3800020109. 2.3, 4.1.1

[10] Wei Liu, Seong-Woo Kim, Zhuang J. Chong, Xiatong Shen, and Marcelo H. Ang. Motion planning using cooperative perception on urban road. In *2013 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 130–137, November 2013. doi: 10.1109/RAM.2013.6758572. 2.1

[11] Aaron Miller, Kyungzun Rim, Parth Chopra, Paritosh Kelkar, and Maxim Likhachev. Cooperative perception and localization for cooperative driving. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020. 1.1, 6.2.3

[12] Eduardo Montijano, Rosario Aragues, and Carlos Sagüés. Distributed data association in robotic networks with cameras and limited communications. *IEEE Transactions on Robotics*, 29(6):1408–1423, 2013. 2.2

[13] Reza Olfati-Saber. Distributed kalman filter with embedded consensus filters. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 8179–8184. IEEE, 2005. 2.2

[14] Reza Olfati-Saber. Distributed kalman filtering for sensor networks. In *2007 46th IEEE Conference on Decision and Control*, pages 5492–5498. IEEE, 2007. 2.2

[15] Reza Olfati-Saber. Kalman-consensus filter: Optimality, stability, and performance. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 7036–7042. IEEE, 2009. 2.2

[16] Jeroen Ploeg, Elham Semsar-Kazerooni, Alejandro I. Morales Medina, Jan F. C. M. de Jongh, Jacco van de Sluis, Alexey Voronov, Cristofer Englund, Reinder J. Bril, Hrishikesh Salunkhe, Álvaro Arrúe, Aitor Ruano, Lorena García-Sol, Ellen van Nunen, and Nathan van de Wouw. Cooperative Automated Maneuvering at the 2016 Grand Cooperative Driving Challenge. *IEEE Transactions on Intelligent Transportation Systems*, 19(4):1213–1226, April 2018. ISSN 1524-9050. doi: 10.1109/TITS.2017.2765669. 2.1

[17] Andreas Rauch, Stefan Maier, Felix Klanner, and Klaus Dietmayer. Inter-vehicle object association for cooperative perception systems. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 893–898, October 2013. doi: 10.1109/ITSC.2013.6728345. 2.1

[18] Suryansh Saxena, Isaac K. Isukapati, Stephen F. Smith, and John M. Dolan. Multiagent sensor fusion for connected & autonomous vehicles to enhance naviga-

tion safety. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2490–2495, 2019. 2.1

[19] Samuel Schulter, Paul Vernaza, Wongun Choi, and Manmohan Chandraker. Deep Network Flow for Multi-Object Tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6951–6960, 2017. 2.3

[20] Sarthak Sharma, Junaid Ahmed Ansari, J. Krishna Murthy, and K. Madhava Krishna. Beyond Pixels: Leveraging Geometry and Shape Cues for Online Multi-Object Tracking. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3508–3515, May 2018. doi: 10.1109/ICRA.2018.8461018. 2.3

[21] Ola Shorinwa, Javier Yu, Trevor Halsted, Alex Koufos, and Mac Schwager. Distributed multi-target tracking for autonomous vehicle fleets. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020. 2.2

[22] Xinshuo Weng and Kris Kitani. A Baseline for 3D Multi-Object Tracking. *arXiv:1907.03961 [cs]*, July 2019. 2.3

[23] Xinshuo Weng, Yongxin Wang, Yunze Man, and Kris M. Kitani. Gnn3dmot: Graph neural network for 3d multi-object tracking with 2d-3d multi-feature learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2.3, 6.2.1

[24] Xinshuo Weng, Ye Yuan, and Kris Kitani. Joint 3d tracking and forecasting with graph neural network and diversity sampling. *arXiv preprint arXiv:2003.07847*, 2020. 2.3, 6.2.1

[25] Zongze Wu, Minyue Fu, Yong Xu, and Renquan Lu. A distributed kalman filtering algorithm with fast finite-time convergence for sensor networks. *Automatica*, 95: 63–72, 2018. 2.2

[26] Yihong Xu, Yutong Ban, Xavier Alameda-Pineda, and Radu Horaud. DeepMOT: A Differentiable Framework for Training Multiple Object Trackers. *arXiv:1906.06618 [cs]*, June 2019. 2.3