2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France
[Award version is not citable, please cite the contributed session version]

Distributed Multi-Target Tracking for Autonomous Vehicle Fleets

Ola Shorinwa[1], Javier Yu[2], Trevor Halsted[1], Alex Koufos[2], and Mac Schwager[2]

***Abstract*— We present a scalable distributed target tracking algorithm based on the alternating direction method of multipliers that is well-suited for a fleet of autonomous cars communicating over a vehicle-to-vehicle network. Each sensing vehicle communicates with its neighbors to execute iterations of a Kalman filter-like update such that each agent's estimate approximates the centralized *maximum a posteriori* estimate without requiring the communication of measurements. We show that our method outperforms the Consensus Kalman Filter in recovering the centralized estimate given a fixed communication bandwidth. We also demonstrate the algorithm in a high fidelity urban driving simulator (CARLA), in which 50 autonomous cars connected on a time-varying communication network track the positions and velocities of 50 target vehicles using on-board cameras.**

I. Introduction

A key challenge in integrating autonomous vehicles into the transportation infrastructure is ensuring their safe operation in the presence of potential hazards, such as human-operated vehicles and pedestrians. However, tracking the paths of these safety-critical targets using on-board sensors is difficult in urban environments due to the presence of occlusions. Collaborative estimation among networked autonomous vehicles has the potential to alleviate the limitations of each vehicle's individual perception capabilities. Networked fleets of autonomous vehicles operating in urban environments can collectively improve the safety of their planning and decision-making by collaboratively tracking the trajectories of nearby vehicles in real-time.

Constraints on communication and computation impose fundamental challenges on collaborative tracking. Given limited communication bandwidth, information communicated between vehicles must be succinct and actionable. Communication channels must also be free to form and dissolve responsively given the highly dynamic nature of urban traffic. Relying on centralized computation is neither robust to single points of failure, nor communication-efficient in disseminating information to those vehicles to whom it is relevant. Rather, a fully-distributed scheme that exploits the computational and communication resources of an autonomous fleet is crucial to reliable tracking.

*This project was funded in part by DARPA YFA award D18AP00064, NSF NRI award 1830402. Toyota Research Institute ("TRI") provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity. The second author was funded on an NSF GRF, and the third on an NDSEG Fellowship.

** The first three authors contributed equally.

[1]Department of Mechanical Engineering, Stanford University, Stanford, CA 94305, USA, {shorinwa, halsted}@stanford.edu

[2]Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94305, USA {javieryu, akoufos, schwager}@stanford.edu

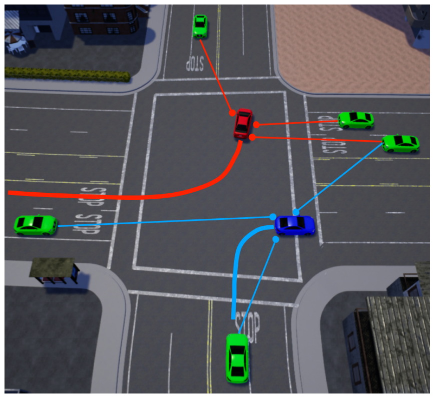

Fig. 1. Autonomous vehicles (in green) track the trajectory of target vehicles (in blue and red) with images from on-board cameras at a four-way intersection using our algorithm.

In this paper, we consider the problem of distributed target tracking in a fleet of vehicles collaborating over a dynamic communication network, posed as a Maximum A Posteriori (MAP) optimization problem. Our key contribution is a scalable Distributed Rolling Window Tracking (DRWT) algorithm derived from the Alternating Direction Method of Multipliers (ADMM) distributed optimization framework. The algorithm consists of closed-form algebraic iterations reminiscent of the Kalman filter and Kalman smoother, but guarantees that the network of vehicles converge to the centralized MAP estimate of the targets' trajectories over a designated sliding time window. We show in extensive simulations that our DRWT algorithm converges to the centralized estimate orders of magnitude faster than a state-of-the art Consensus Kalman Filter for the same bandwidth. We demonstrate our algorithm in a realistic urban driving scenario in the CARLA simulator, in which 50 autonomous cars track 50 target vehicles in real time using only segmented images from their on-board cameras.

The paper is organized as follows. We give related work in Sec. II and pose the distributed estimation problem in Sec. III. In Sec. IV, we formulate the centralized MAP optimization problem, and we derive our DRWT algorithm in Sec. V. Sec. VI presents results comparing our DRWT to the Consensus Kalman Filter, and describes large-scale simulations in a CARLA urban driving scenario.

II. Related Work

Several approaches have previously been applied to solving distributed estimation problems. In distributed filtering methods, consensus techniques enable the asymptotic diffusion of information throughout the communication network, allowing individual computation nodes to approximate the joint estimate in the Consensus Kalman Filter [1], [2], [3], [4]. Alternatively, using finite consensus techniques can improve communication efficiency [5]. Similar techniques have also been applied to particle filtering [6], [7]. However, the messages communicated in these consensus-based methods contain both information vectors and information matrices, so the communication cost scales superlinearly with the size of the estimate. Our approach recovers the same centralized solution while only communicating the estimate vector.

Sensor fusion techniques accomplish distributed estimation by computing a centralized approximation given individual estimates throughout the network [8]. A key challenge in sensor fusion is keeping track of cross-correlation in conditioning individual estimates on previously-fused centralized estimates [9]. Covariance Intersection (CI) addresses this issue by computing a consistent centralized estimate that accounts for any possible cross-correlation between individual estimates [10], [11], [12], [13]. However, in ensuring consistency, CI is often extremely conservative and therefore significantly suboptimal, especially for large networks.

Other estimation techniques approach distributed estimation using optimization. One approach is to aggregate all observations of each target to form a non-linear least squares objective function which recovers the MAP estimate [14], though such an approach requires all-to-all communication. In [15], each robot communicates its measurement and state estimate to its neighbors to solve the MAP least-squares problem using the conjugate gradient method. However, this approach still requires each node to communicate its measurements to its neighbors. Alternatively, some methods have been proposed to divide targets among the trackers using Voronoi partitions [16], and to track multiple targets using the Probability Hypothesis Density (PHD) filter [17].

In this paper, we apply a novel approach to the problem of target tracking. We pose target tracking as a MAP estimate over a rolling window that bears some similarity to [18]. We apply ADMM, a technique that allows for distributed optimization of problems with separable objectives, to distribute the resulting MAP optimization problem (see [19], [20] for a detailed survey of ADMM). This approach guarantees convergence to the centralized solution [21].

III. Problem Formulation

A. Communication model

We consider the scenario of N camera-equipped autonomous vehicles ("sensors") navigating a city that also contains M other vehicles ("targets"). Each sensor takes measurements of the positions of the targets in its vicinity and can communicate with other nearby sensors. We model the communication network among the N sensors at time t as a dynamic undirected graph $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}_t)$, with vertices $\mathcal{V} = \{1, \dots, N\}$ corresponding to sensors and edges $\mathcal{E}_t$ containing pairs of sensors that can directly share information with each other. The presence of an edge (i, j) depends on the proximity between sensors i and j at time t. The neighbor set $\mathcal{N}_{i,t} = \{j \mid (i, j) \in \mathcal{E}_t\}$ consists of sensors j that can communicate with sensor i at time t.

B. Target assignment

We assume that the each target in the environment has a unique identifier known to all sensors. This data association task is addressed in [22], and can be performed in a completely distributed fashion.

The set of sensors observing any given target changes due to occlusions coupled with the limited sensing-range of the cameras. At each time that a sensor observes one or more targets, it generates a set of features for each target ([23], [24], [25]) which identify the target. This identifier is communicated to its neighbors. Considering the case of a particular target, we denote the set of sensors that observe it over the time horizon $[t - T, t]$ as $\mathcal{W}_t$. The subgraph of sensors that are relevant to the target in the time horizon is $\mathcal{G}'_t \subseteq \mathcal{G}_t$, such that $\mathcal{V}'_t = \mathcal{V} \cap \mathcal{W}_t$ and $\mathcal{E}'_t = \{(i, j) \mid (i, j) \in \mathcal{E}_t, i, j \in \mathcal{V}'_t\}$. Sensor i knows that sensor j belongs to $\mathcal{V}'_t$ since sensor j communicates a descriptor of each observed target. We assume that the subgraph $\mathcal{G}'_t$ is connected at all times t (that is, there exists a set of edges that form a path between any $i, j \in \mathcal{V}'_t$).

C. Distributed estimation

Given a particular target, each sensor has the task of estimating the target's state $\mathbf{x} \in \mathbb{R}^n$ which includes its position and velocity over discrete timesteps modeled as a linear Gaussian system in which

$$\mathbf{x}_{t+1} = \mathbf{A}_t \mathbf{x}_t + \mathbf{w}_t, \tag{1}$$

with linear dynamics $\mathbf{A}_t \in \mathbb{R}^{n \times n}$ and additive noise $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_t) \in \mathbb{R}^n$. In the following, we represent the trajectory over the time horizon $[t - T, t]$ using the notation $\mathbf{x}_{t-T:t} = \begin{bmatrix} \mathbf{x}_{t-T}^\top & \cdots & \mathbf{x}_t^\top \end{bmatrix}^\top$. Sensor i makes an observation of the target at time t according to

$$\mathbf{y}_{i,t} = \mathbf{C}_{i,t} \mathbf{x}_t + \mathbf{v}_{i,t}, \tag{2}$$

with measurement vector $\mathbf{y}_{i,t} \in \mathbb{R}^{m_i}$, measurement matrix $\mathbf{C}_{i,t} \in \mathbb{R}^{m_i \times n}$, and additive noise $\mathbf{v}_{i,t} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{i,t}) \in \mathbb{R}^{m_i}$. We also refer to the joint set of observations across all sensors in the network as

$$\mathbf{y}_t = \mathbf{C}_t \mathbf{x}_t + \mathbf{v}_t, \tag{3}$$

where the joint variables $\mathbf{y}_t \in \mathbb{R}^m$, $\mathbf{C}_t \in \mathbb{R}^{m \times n}$, and $\mathbf{v}_t \in \mathbb{R}^m$ are the column-wise concatenations over all $i \in \mathcal{V}'_t$, of $\mathbf{y}_{i,t}$, $\mathbf{C}_{i,t}$ and $\mathbf{v}_{i,t}$, respectively.

While the joint measurements (3) are not available to any single sensing agent, each agent uses its individual measurements (2) as well as communication with its neighbors to estimate the target's state. We compare the sensor's estimated mean and covariance with the mean and covariance computed with full knowledge of all measurements. In the *distributed estimation problem*, each sensor seeks to approximate the centralized (best-possible) estimate using only individual measurements and local communication.

IV. Centralized Estimation

The centralized estimate, which is conditioned on all measurements and priors in the network, gives the best estimate of a target's state and therefore represents the best possible performance. The MAP batch estimate maximizes the probability of the estimated target trajectory $\mathbf{x}_{0:t}$ conditioned on the full set of measurements $\mathbf{y}_{0:t}$ and a prior of mean $\bar{\mathbf{x}}_0$ and covariance $\bar{\mathbf{P}}_0$:

$$\hat{\mathbf{x}}_{0:t} = \arg\max_{\mathbf{x}_{0:t}} p(\mathbf{x}_{0:t} \mid \bar{\mathbf{x}}_0, \mathbf{y}_{1:t}) \tag{4}$$

$$= \arg\max_{\mathbf{x}_{0:t}} p(\mathbf{x}_0 \mid \bar{\mathbf{x}}_0) \prod_{\tau=0}^{t-1} p(\mathbf{x}_{\tau+1} \mid \mathbf{x}_\tau) \prod_{\tau=1}^{t} p(\mathbf{y}_\tau \mid \mathbf{x}_\tau). \tag{5}$$

Given Gaussian conditional probabilities, the posterior in (5) is the Gaussian distribution $\mathcal{N}(\hat{\mathbf{x}}_{0:t}, \hat{\mathbf{P}}_{0:t})$.

In the case of linear Gaussian systems, we can solve (5) as a linear system of equations. However, recursively estimating the trajectory reduces the size of the system of equations, improving computational efficiency. Instead of maximizing $p(\mathbf{x}_t \mid \bar{\mathbf{x}}_0, \mathbf{y}_{0:t})$, the Kalman filter infers $\hat{\mathbf{x}}_t$ from the result of the previous timestep's estimate, the prior distribution $(\bar{\mathbf{x}}_{t-1}, \bar{\mathbf{P}}_{t-1})$:

$$\hat{\mathbf{x}}_t = \arg\max_{\mathbf{x}_t} p(\mathbf{x}_t \mid \bar{\mathbf{x}}_{t-1}, \mathbf{y}_t). \tag{6}$$

However, the Kalman filter only exactly replicates the result of the batch estimate for the final timestep t. For some intermediate $\tau < t$, $\hat{\mathbf{x}}_\tau$ is conditioned on the full measurement set $\mathbf{y}_{0:t}$ in the batch approach, but only on $\mathbf{y}_{0:\tau}$ in the filtering approach. Employing the Rauch-Tung-Striebel smoother exactly recovers the batch solution by computing $p(\hat{\mathbf{x}}_\tau \mid \hat{\mathbf{x}}_{\tau+1})$ for $\tau = t-1, \ldots, 0$ (a backward pass of the trajectory performed after the Kalman filter's forward pass).

For our application of persistently tracking targets, a MAP rolling window approach is appropriate as it incorporates smoothing effects into a single Kalman filter-like update. The rolling window refers to a time horizon $[t-T, t]$ over which we compute the MAP estimate. Given the prior $(\bar{\mathbf{x}}_{t-T:t-1}, \bar{\mathbf{P}}_{t-T:t-1})$, we compute the window's posterior distribution by factoring the original MAP solution as

$$\hat{\mathbf{x}}_{t-T:t} = \arg\max_{\mathbf{x}_{t-T:t}} \big\{ p(\mathbf{x}_{t-T:t-1} \mid \bar{\mathbf{x}}_{t-T:t-1}) \\ p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) p(\mathbf{y}_t \mid \mathbf{x}_t) \big\}. \tag{7}$$

The estimate $\hat{\mathbf{x}}_{t-T:t}$ is conditioned on $\bar{\mathbf{x}}_0$, $\mathbf{y}_{0:t}$ and is equivalent to performing a filtering pass for the times $0, \ldots, t$ and a smoothing pass from time t to time $t-T$. We then increment the rolling window forward to $[t-T+1, t+1]$, retaining the estimate $(\hat{\mathbf{x}}_{t-T+1:t}, \hat{\mathbf{P}}_{t-T+1:t})$ as the prior for that window. Therefore, the rolling window approach preserves much of the smoothing effect of the batch estimate while maintaining a constant problem size at each time step.

Applying (1) and (3) to (7) yields

$$J(\hat{\mathbf{x}}_{t-T:t}) = \|\hat{\mathbf{x}}_t - \mathbf{A}_{t-1}\hat{\mathbf{x}}_{t-1}\|^2_{\mathbf{Q}_{t-1}^{-1}} + \|\mathbf{y}_t - \mathbf{C}_t\hat{\mathbf{x}}_t\|^2_{\mathbf{R}_t^{-1}} \\ + \|\hat{\mathbf{x}}_{t-T:t-1} - \bar{\mathbf{x}}_{t-T:t-1}\|^2_{\bar{\mathbf{P}}_{t-T:t-1}^{-1}}. \tag{8}$$

for which the minimizing $\hat{\mathbf{x}}_{t-T:t}$ is the solution to (7).

We can express the MAP rolling window estimate as

$$\hat{\mathbf{x}}_{t-T:t} = \left(\mathbf{H}_t^\top \mathbf{W}_t^{-1} \mathbf{H}_t\right)^{-1} \mathbf{H}_t^\top \mathbf{W}_t^{-1} \mathbf{z}_t \tag{9}$$

$$\hat{\mathbf{P}}_{t-T:t} = \left(\mathbf{H}_t^\top \mathbf{W}_t^{-1} \mathbf{H}_t\right)^{-1}, \tag{10}$$

given the block matrices

$$\mathbf{H}_t = \left[\begin{array}{cccc} \mathbf{0} & \cdots & -\mathbf{A}_{t-1} & \mathbf{I} \\ \hline \mathbf{0} & \cdots & \mathbf{0} & \mathbf{C}_t \\ \hline \mathbf{I} & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \cdots & \mathbf{I} & \mathbf{0} \end{array}\right] = \begin{bmatrix} \mathbf{F}_t \\ \mathbf{G}_t \\ \mathbf{\Pi}_t \end{bmatrix},$$

$$\mathbf{z}_t = \begin{bmatrix} \mathbf{0}^\top & \mathbf{y}_t^\top & \bar{\mathbf{x}}_{t-T:t-1}^\top \end{bmatrix}^\top,$$

$$\mathbf{W}_t = \text{blkdiag}\left(\mathbf{Q}_{t-1}, \mathbf{R}_t, \hat{\mathbf{P}}_{t-T:t-1}\right).$$

We implement this procedure recursively by retaining the lower-right block of the covariance matrix $\hat{\mathbf{P}}_{t-T:t}$ as the prior covariance $\bar{\mathbf{P}}_{t-T+1:t}$ for the next timestep's estimate. The estimate over all but timestep $t-T$ becomes the prior mean $\bar{\mathbf{x}}_{t-T+1:t}$. Therefore, we have a tractable centralized target tracking method that serves as a benchmark for our distributed target tracking algorithm.

V. Distributed Estimation

One typical approach for the distributed implementation of the MAP estimate is to use consensus techniques to diffuse information across the network, enabling each agent to minimize (8). This is true of Consensus Kalman Filter (CKF) approaches, in which each agent maintains local measurement information (2) rather than the joint measurements [2], [3], [4], [1], [5]. The CKF uses asymptotic consensus with Metropolis weights to sum $\mathbf{G}_{i,t}^\top \mathbf{R}_{i,t}^{-1} \mathbf{G}_{i,t}$ and $\mathbf{G}_{i,t}^\top \mathbf{R}_{i,t}^{-1} \mathbf{y}_{i,t}$ over all $i \in \mathcal{V}_t'$, where $\mathbf{G}_{i,t} = \begin{bmatrix} \mathbf{0} & \ldots & \mathbf{0} & \mathbf{C}_{i,t} \end{bmatrix}$. The fused observations are then fused with local copies of the dynamics terms and prior terms of the cost function. The consensus rounds diffuse the joint measurement information to each sensor, enabling local computation of (9) and (10).

The CKF requires communication of local information matrices and information vectors during consensus, a communication-intensive process that is a drawback of the method. Furthermore, performing an approximation of the centralized estimate at each node is redundant, failing to take advantage of the distributed nature of the computational resources in the network. In contrast to the CKF, we propose a Distributed Rolling Window Tracking (DRWT) algorithm that uses an ADMM-based approach to enable each sensor to replicate the centralized estimate without reconstructing the centralized cost function. First, we pose the centralized cost function (8) as a separable problem with linear constraints:

$$\begin{aligned}\underset{\hat{\mathbf{x}}_{i,t-T:t}\forall i\in\mathcal{V}'_t}{\text{minimize}} \sum_{i\in\mathcal{V}'_t} \Bigg\{ & \frac{1}{|\mathcal{V}'_t|} \|\hat{\mathbf{x}}_{i,t} - \mathbf{A}_{t-1}\hat{\mathbf{x}}_{i,t-1}\|^2_{\mathbf{Q}^{-1}_{t-1}} \\ & + \|\mathbf{y}_{i,t} - \mathbf{C}_{i,t}\hat{\mathbf{x}}_{i,t}\|^2_{\mathbf{R}^{-1}_{i,t}} \\ & + \|\hat{\mathbf{x}}_{i,t-T:t-1} - \bar{\mathbf{x}}_{i,t-T:t-1}\|^2_{\bar{\mathbf{P}}^{-1}_{i,t-T:t-1}} \Bigg\} \\ \text{subject to} \quad & \hat{\mathbf{x}}_{i,t-T:t} = \mathbf{r}_{ij} \qquad \forall j \in \mathcal{N}_{i,t} \\ & \hat{\mathbf{x}}_{j,t-T:t} = \mathbf{r}_{ij} \qquad \forall j \in \mathcal{N}_{i,t},\end{aligned} \tag{11}$$

for which $\sum_{i\in\mathcal{V}'_t} \bar{\mathbf{P}}^{-1}_{i,t-T:t-1} = \bar{\mathbf{P}}^{-1}_{t-T:t-1}$. In the following, we express the cost function in (11) as $\sum_{i\in\mathcal{V}'_t} J_i(\mathbf{x}_{i,t-T:t})$ and omit the subscript $t-T:t$ from the primal variable $\hat{\mathbf{x}}_i$. The slack variable $\mathbf{r}_{ij} \in \mathbb{R}^{n(T+1)}$ encodes agreement constraints between neighbors i and j. The ADMM approach to solving problems of this form uses the augmented Lagrangian, which adds to the cost function a quadratic penalty for constraint violations, $\sum_{i\in\mathcal{V}'_t}\sum_{j\in\mathcal{N}_{i,t}}(\rho/2)(\|\mathbf{x}_i - \mathbf{r}_{ij}\|^2 + \|\mathbf{x}_j - \mathbf{r}_{ij}\|^2)$. The augmented problem is equivalent to the original problem as the added penalty is zero for the feasible set of estimates. We find the saddle point of the augmented Lagrangian

$$\begin{aligned} L_\rho = \sum_{i\in\mathcal{V}'_t} J_i(\mathbf{x}_i) + & \sum_{j\in\mathcal{N}_{i,t}} \left(\lambda_{ij}^\top(\mathbf{x}_i - \mathbf{r}_{ij}) + \mu_{ij}^\top(\mathbf{x}_j - \mathbf{r}_{ij})\right) \\ & + \frac{\rho}{2}\sum_{j\in\mathcal{N}_{i,t}}\left(\|\mathbf{x}_i - \mathbf{r}_{ij}\|^2 + \|\mathbf{x}_j - \mathbf{r}_{ij}\|^2\right) \end{aligned} \tag{12}$$

by alternating between minimizing L_ρ with respect to the primal variables $\mathbf{x}$ and $\mathbf{r}$ and performing a gradient ascent step on the dual variables λ_{ij} and μ_{ij}. Each $i \in \mathcal{V}'_t$ can update its respective $\mathbf{x}_i$, $\mathbf{r}_{ij}$, λ_{ij}, and μ_{ij} for all $j \in \mathcal{N}_{i,t}$ in parallel since minimizing L_ρ with respect to these variables does not depend on the values of its neighbors' variables. Furthermore, as shown in [20], [26], substituting $\mathbf{p}_i = \sum_{j\in\mathcal{N}_{i,t}} \lambda_{ij} + \mu_{ij}$ and assuming the initialization $\mathbf{p}_i^{(0)} = \mathbf{0}$ yields the minimization of $\mathbf{r}_{ij}$ as $\frac{1}{2}(\mathbf{x}_i + \mathbf{x}_j)$. Initializing $\mathbf{p}_i^{(0)} = \mathbf{0}$ and $\hat{\mathbf{x}}_i{}^{(0)} = \arg\min_{\mathbf{x}_i} J_i(\mathbf{x}_i)$, the following iterations alternate between a gradient ascent step on $\mathbf{p}_i$ and a minimization step on $\mathbf{x}_i$, converging to the centralized estimate when run in parallel across all $i \in \mathcal{V}'_t$:

$$\mathbf{p}_i^{(k+1)} = \mathbf{p}_i^{(k)} + \rho \sum_{j\in\mathcal{N}_{i,t}} \left(\hat{\mathbf{x}}_i^{(k)} - \hat{\mathbf{x}}_j^{(k)}\right) \tag{13}$$

$$\begin{aligned}\hat{\mathbf{x}}_i^{(k+1)} = \arg\min_{\mathbf{x}_i} \Bigg\{ & J_i(\mathbf{x}_i) + \mathbf{x}_i^\top \mathbf{p}_i^{(k+1)} \\ & + \rho \sum_{j\in\mathcal{N}_{i,t}} \left\|\mathbf{x}_i - \frac{1}{2}\left(\hat{\mathbf{x}}_i^{(k)} + \hat{\mathbf{x}}_j^{(k)}\right)\right\|^2 \Bigg\}\end{aligned} \tag{14}$$

Furthermore, due to our assumption of a linear Gaussian system (14) can be expressed in closed form as

$$\begin{aligned}\left(\mathbf{H}_i^\top \mathbf{W}_i^{-1}\mathbf{H}_i + 2\rho|\mathcal{N}_i|\mathbf{I}\right)\hat{\mathbf{x}}_i^{(k+1)} = & \\ \mathbf{H}_i^\top \mathbf{W}_i^{-1}\mathbf{z}_i - \mathbf{p}_i^{(k+1)} + \rho & \sum_{j\in\mathcal{N}_i}\left(\hat{\mathbf{x}}_i^{(k)} + \hat{\mathbf{x}}_j^{(k)}\right).\end{aligned} \tag{15}$$

using the local versions of the block matrices in (9) (replacing $\mathbf{C}_t$, $\mathbf{Q}_{t-1}$, $\mathbf{R}_t$, $\bar{\mathbf{P}}_{t-T:t-1}$, $\mathbf{y}_t$, and $\bar{\mathbf{x}}_{t-T:t-1}$ with $\mathbf{C}_{i,t}$, $\mathbf{Q}_t/|\mathcal{V}'_t|$, $\mathbf{R}_{i,t}$, $\bar{\mathbf{P}}_{i,t-T:t-1}$, $\mathbf{y}_{i,t}$, and $\bar{\mathbf{x}}_{i,t-T:t-1}$, respectively). We note that the matrix inverse in (15) only needs to be computed once rather than at every primal update iteration.

Lemma 1. *Given a connected $\mathcal{G}'_t$ and priors $\bar{\mathbf{x}}_{i,t-T:t-1}$ and $\bar{\mathbf{P}}_{i,t-T:t-1}$ such that $\bar{\mathbf{x}}_{i,t-T:t-1} = \bar{\mathbf{x}}_{t-T:t-1} \forall i \in \mathcal{V}'_t$ and $\sum_{i\in\mathcal{V}'_t} \bar{\mathbf{P}}^{-1}_{i,t-T:t-1} = \bar{\mathbf{P}}^{-1}_{t-T:t-1}$, there is a saddle point of (12) at*

$$\hat{\mathbf{x}}_i^{(k)} = \hat{\mathbf{x}}_{t-T:t} \tag{16}$$

$$\mathbf{p}_i^{(k)} = \mathbf{H}_i^\top \mathbf{W}_i^{-1}\mathbf{z}_i - \mathbf{H}_i^\top \mathbf{W}_i^{-1}\mathbf{H}_i\hat{\mathbf{x}}_{t-T:t}, \tag{17}$$

where $\hat{\mathbf{x}}_{t-T:t}$ is the centralized MAP rolling window estimate given priors $\bar{\mathbf{x}}_{i,t-T:t-1}$, $\bar{\mathbf{P}}_{i,t-T:t-1}$.

Proof. The Hessian of L_ρ with respect to the primal variables $\hat{\mathbf{x}}_i \, \forall i \in \mathcal{V}'_t$ is positive definite. Observing that

$$\left.\frac{\partial L_\rho}{\partial \hat{\mathbf{x}}_i}\right|_{\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_{t-T:t}} = \frac{\partial}{\partial \hat{\mathbf{x}}_i} J(\hat{\mathbf{x}}_{t-T:t}) = \frac{\partial}{\partial \hat{\mathbf{x}}_i} J(\hat{\mathbf{x}}_{t-T:t}) = 0$$

for each $i \in \mathcal{V}'_t$, L_ρ is minimized with respect to the primal variables at $\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_{t-T:t} \, \forall i \in \mathcal{V}'_t$. Substituting the primal solution into the dual update, we see that $\frac{\partial L_\rho}{\partial \mathbf{p}_i} = 0$. Substituting (16) into (15) yields (17). □

In other words, the network can minimize (8) in a fully distributed manner using only independent measurements and local communication. By decomposing the centralized problem according to (11), each estimate $\hat{\mathbf{x}}_i$ converges to the solution of (9). A key assumption, however, is the decomposability of prior information, *i.e.*, $\sum_{i\in\mathcal{V}'_t} \bar{\mathbf{P}}^{-1}_{i,t-T:t-1} = \bar{\mathbf{P}}^{-1}_{t-T:t-1}$. Given that the distributed prior inverse covariances sum to the centralized prior inverse covariances, then the distributed posterior inverse covariances (where $\hat{\mathbf{P}}_{t-T:t}$ is the Hessian of the local cost function J_i) also sum to the centralized posterior inverse covariances. However, this assumption weakens in implementing DRWT recursively. In performing the marginalization step in which $\bar{\mathbf{P}}_{t-T+1:t}$ is the $t-T+1:t$ block of $\hat{\mathbf{P}}_{t-T:t}$, the distributed implementation is not exactly equivalent to the centralized. It is always true that $(\sum_{i\in\mathcal{V}'_t} \bar{\mathbf{P}}^{-1}_{i,t-T:t-1})^{-1} \geq \bar{\mathbf{P}}_{t-T:t-1}$. Consequently, the distributed marginalization is conservative with respect to the centralized solution. The conservativeness of the estimated covariance is a feature of other distributed algorithms as well—as Figure 3 shows, the CKF has an even more conservative covariance estimate. Therefore, while DRWT remains an unbiased estimator, it does not exactly replicate the centralized covariance in its recursive implementation, as the prior mean is under-weighted. Lemma 1 holds, with the modification that the saddle point is the solution to a centralized optimization problem with a potentially overestimated prior covariance. As we show in Sec. VI, this effect is minimal in practice.

Finally, we propose a "hand-off" protocol by which sensor i removes itself from estimating a target after not directly observing it in the T most recent timesteps. If there exists $j \in \mathcal{N}_{i,t} \cap \mathcal{V}'_{t+1}$ (*i.e.*, neighbor j is continuing to estimate

Algorithm 1 Distributed Rolling Window Tracking

```
 1: function DRWT($\bar{\mathbf{x}}_{i,t-T:t-1}, \bar{\mathbf{P}}_{i,t-T:t-1}, \mathbf{y}_{i,t} \quad \forall i \in \mathcal{V}_t'$)
 2:     for $i \in \mathcal{V}_t'$ do
 3:         $\hat{\mathbf{x}}_{i,t-T:t}^{(0)} \leftarrow \arg\min_{\mathbf{x}_{i,t-T:t}} J_i(\mathbf{x}_{i,t-T:t})$
 4:         $\mathbf{p}_i^{(0)} \leftarrow \mathbf{0}$
 5:         $\hat{\mathbf{P}}_{i,t-T:t} \leftarrow \left(\mathbf{H}_{i,t}^\top \mathbf{W}_{i,t}^{-1} \mathbf{H}_{i,t}\right)^{-1}$
 6:     end for
 7:     while stopping criterion is unmet do
 8:         for $i \in \mathcal{V}_t'$ do
 9:             $\mathbf{p}_i^{(k+1)} \leftarrow$ Equation (13)          ▷ dual update
10:             $\hat{\mathbf{x}}_{i,t-T:t}^{(k+1)} \leftarrow$ Equation (14)      ▷ primal update
11:         end for
12:         $k \leftarrow k+1$
13:     end while
14:     for $i \in \mathcal{V}_t' \notin \mathcal{V}_{t+1}', j \in \mathcal{N}_{i,t} \cap \mathcal{V}_{t+1}'$ do
15:         $\hat{\mathbf{P}}_{j,t-T:t} \leftarrow \left(\hat{\mathbf{P}}_{i,t-T:t}^{-1} + \hat{\mathbf{P}}_{j,t-T:t}^{-1}\right)^{-1}$   ▷ hand-off
16:     end for
17:     return $\hat{\mathbf{x}}_{i,t-T:t}, \hat{\mathbf{P}}_{i,t-T:t} \quad \forall i \in \mathcal{V}_t'$
18: end function
```

Algorithm 2 DRWT Primal Update

```
 1: function PrimalUpdate($\mathbf{p}_i^{(k)}, \hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j \ \forall j \in \mathcal{N}_{i,t}$)
 2:     $\boldsymbol{\alpha}_\tau := \frac{\rho}{2} \sum_{j \in \mathcal{N}_i} \left(\hat{\mathbf{x}}_{i,\tau}^{(k)} + \hat{\mathbf{x}}_{j,\tau}^{(k)}\right) - \frac{1}{2}\mathbf{p}_{i,\tau}^{(k)}$
 3:     $\boldsymbol{\gamma}_\tau := \frac{1}{|\mathcal{V}_t'|}\mathbf{Q}_{\tau-1}^{-1} + \rho\,|\mathcal{N}_i|$
 4:     initialization
 5:         $\boldsymbol{\Phi}_{T-t} \leftarrow \hat{\mathbf{P}}_{i,T-t}^{-1}\bar{\mathbf{x}}_{i,T-t} + \rho\,|\mathcal{N}_i|$
 6:         $\boldsymbol{\beta}_{i,T-t} \leftarrow \hat{\mathbf{P}}_{i,T-t}^{-1}\bar{\mathbf{x}}_{i,T-t} + \boldsymbol{\alpha}_{T-t}$
 7:     forward pass
 8:         for $\tau = t-T+1, \cdots, t$ do
 9:             $\mathbf{L}_{\tau-1}\mathbf{L}_{\tau-1}^\top \leftarrow \boldsymbol{\Phi}_{\tau-1} + \frac{1}{|\mathcal{V}_t'|}\mathbf{A}_{\tau-1}^\top \mathbf{Q}_{\tau-1}^{-1}\mathbf{A}_{\tau-1}$
10:             $\mathbf{L}_{\tau,\tau-1}\mathbf{L}_{\tau-1}^\top \leftarrow -\frac{1}{|\mathcal{V}_t'|}\mathbf{Q}_{\tau-1}^{-1}\mathbf{A}_{\tau-1}$
11:             $\mathbf{L}_{\tau-1}\boldsymbol{\sigma}_{\tau-1} \leftarrow \boldsymbol{\beta}_{i,\tau-1}$
12:             $\boldsymbol{\Phi}_\tau \leftarrow -\mathbf{L}_{\tau,\tau-1}\mathbf{L}_{\tau,\tau-1}^\top + \hat{\mathbf{P}}_{i,\tau}^{-1} + \boldsymbol{\gamma}_\tau$
13:             $\boldsymbol{\beta}_{i,\tau} \leftarrow -\mathbf{L}_{\tau,\tau-1}\boldsymbol{\sigma}_{\tau-1} + \hat{\mathbf{P}}_{i,\tau}^{-1}\bar{\mathbf{x}}_{i,\tau} + \boldsymbol{\alpha}_\tau$
14:         end for
15:         $\mathbf{L}_t\mathbf{L}_t^\top \leftarrow -\mathbf{L}_{t,t-1}\mathbf{L}_{t,t-1}^\top + \mathbf{C}_t^{\mathbf{T}}\mathbf{R}_{i,t}^{-1}\mathbf{C}_t + \boldsymbol{\gamma}_t$
16:         $\mathbf{L}_t\boldsymbol{\sigma}_t \leftarrow -\mathbf{L}_{t,t-1}\boldsymbol{\sigma}_{t-1} + \mathbf{C}_t^{\mathbf{T}}\mathbf{R}_{i,t}^{-1}\mathbf{y}_{i,t} + \boldsymbol{\alpha}_t$
17:         $\hat{\mathbf{x}}_{i,t}^{(k+1)} \leftarrow -\mathbf{L}_t^{-\top}\boldsymbol{\sigma}_t$
18:     backward pass
19:         for $\tau = t, \cdots, t-T+1$ do
20:             $\hat{\mathbf{x}}_{i,\tau-1}^{(k+1)} \leftarrow -\mathbf{L}_{\tau-1}^{-\top}\left(\mathbf{L}_{\tau,\tau-1}\hat{\mathbf{x}}_{i,\tau}^{(k+1)} + \boldsymbol{\sigma}_{\tau-1}\right)$
21:         end for
22:     return $\hat{\mathbf{x}}_{i,t-T:t}^{(k+1)}$
23: end function
```

the target), then i transfers the Hessian of its local cost function to a single neighbor j at the end of the ADMM iterations. Sensor j fuses the new information matrix with its own, thereby preserving the same joint information across the entire network. Algorithm 1 summarizes DRWT, including the hand-off protocol.

After each communication round per timestep, sensor i updates its estimate of the target's trajectory (14) by inverting the Hessian of its local objective function which requires $O(n^3(T+1)^3)$ floating point operations (flops), posing a bottleneck for long window lengths. Here, we provide an efficient algorithm for performing this update in $O(n(T+1))$ flops rather than cubic complexity, without any matrix inversion. We factor the Hessian using Cholesky decomposition to obtain a lower triangular matrix $\mathbf{L}_\tau$ for each $\tau = t-T, \cdots, t$ and compute $\boldsymbol{\sigma}_\tau$ to update $\hat{\mathbf{x}}_{i,t-1:t}$ using forward and backward iterations, reminiscent of the Kalman smoothing procedure. The Cholesky decomposition of the Hessian takes $O(n(T+1))$ flops, along with the forward and backward iterations. We present the algorithm in Algorithm 2.

VI. Simulation Results

A. Performance Comparison

We compare the performance of the DRWT method in Algorithm 1 to the CKF in a distributed estimation problem involving a static network with $|\mathcal{V}| = 100$ and $|\mathcal{E}| = 400$. All sensors acquire noisy measurements of the target at each time step, and perform DRWT with $T = 1$. During each estimation phase, the same bandwidth limitations are imposed on the CKF and DRWT. We benchmark both distributed methods against the centralized MAP estimate.

Results from 2000 Monte Carlo simulations of this scenario show that DRWT method outperforms the CKF. DRWT is significantly more communication-efficient, as sensors communicate only their target estimates. From Figure 2, DRWT yields better convergence to the centralized estimate compared to the CKF method as a function of the total number of communication bits per node. As Figure 3 shows, the improved convergence of the DRWT contributes to improved estimation performance over entire trajectories. The estimated trajectories and covariances of the DRWT method closely match the centralized estimates. The CKF does not track the centralized estimate as closely and is also more significantly overconservative in its estimate.

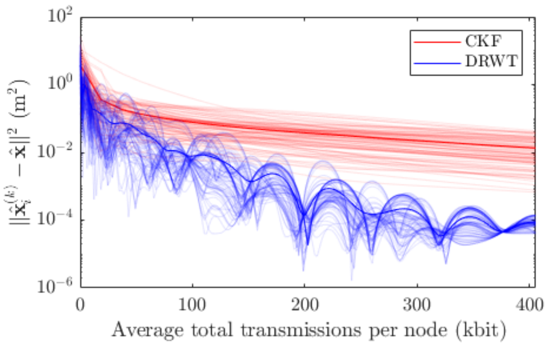

Fig. 2. Convergence of distributed estimation methods to the centralized estimate as a function of bits of communication passed on a 100 node, 400 edge network for a single timestep's estimate.

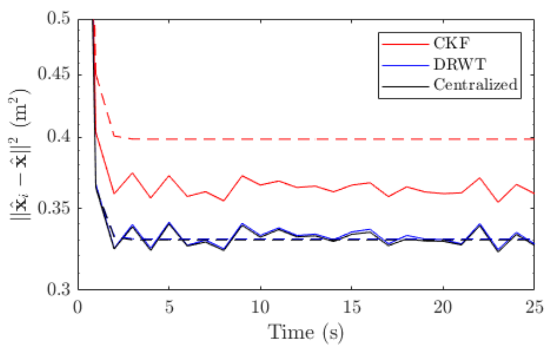

Fig. 3. Mean squared error of estimation methods on a 100 node, 400 edge network with respect to ground truth, averaged over 4000 Monte Carlo simulations. Solid lines show the indicate mean squared error, while dashed lines represent estimated covariances, computed as trace($\hat{\mathbf{P}}$).

B. CARLA Simulations

We demonstrate our algorithm in a scenario involving a network of 50 sensor vehicles and 50 target vehicles within CARLA [27], a simulation test-bed for autonomous driving systems. For the simulation trials, each sensor vehicle is equipped with a forward and a backward-facing camera, each with a 90° field of view. As shown in Figure 4, sensor vehicles acquire semantic segmentation and depth images at 4 Hz. The sensing radius of the vehicles is limited to 100m.

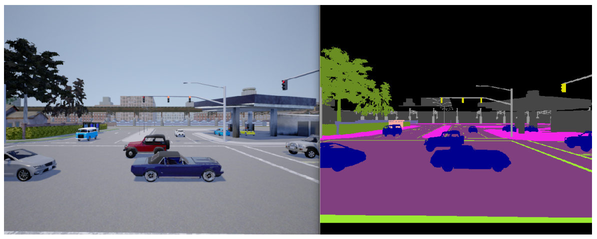

Fig. 4. CARLA frame showing raw and segmented camera images.

The relative position of each target vehicle is deduced from the depth and segmentation images and the camera's projection matrix. Each sensor uses its odometry information to transform the relative position of the target into the global coordinate frame corresponding to the measurement used by the vehicle in DRWT. The sensor estimates trajectories of $T = 5$s in length. For this simulation, we assume that the target labeling is known *a priori*. The communication network between sensor vehicles is modeled as a disk graph with a 200m radius and is updated at 4 Hz. DRWT uses a simple double integrator model for the vehicle dynamics.

Figure 5 shows the mean squared error of the estimated target trajectories of all target vehicles for all the sensor vehicles with respect to the centralized trajectory estimate. Collaborative target tracking using DRWT significantly outperforms the estimates made by any single agent. Increasing the number of iterations of DRWT in each estimation round can further reduce the remaining error.

Figure 6 shows how the information (represented as the trace of the inverse covariance) corresponding to a given target is apportioned across the network. As the set of sensors tracking a target changes in time, the hand-off procedure enables their joint information to closely match the information of the centralized estimate.

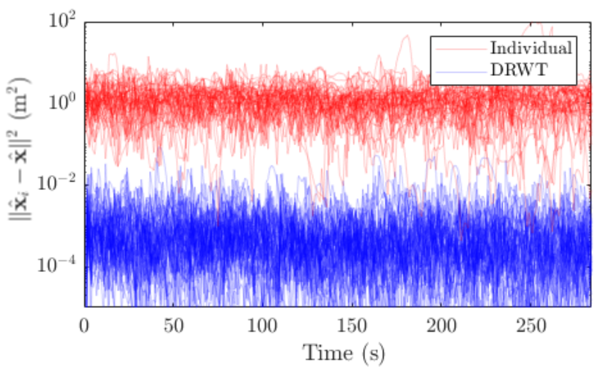

Fig. 5. Mean squared error to the centralized estimate across the full trajectories of all 50 targets. Red lines are the positional estimate errors for each individual sensor (with no communication), and the blue lines are for the DWRT positional estimates.

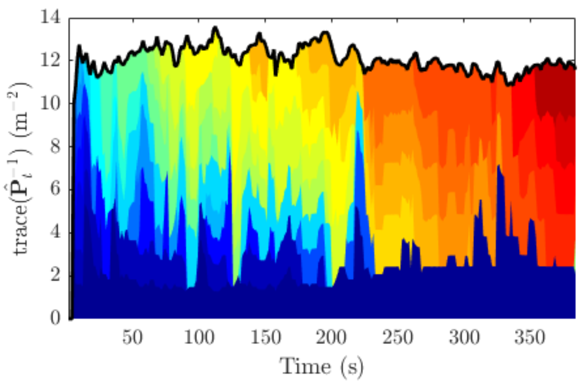

Fig. 6. The sum of the traces of information matrices maintained by sensor vehicles using DRWT for a single target in a CARLA simulation. Each colored band represents the information of one sensor. Although any one sensor possesses only a fraction of the joint information, the sum over the network closely matches the information of a centralized estimator. Spikes in individual bands correspond to execution of the hand-off procedure.

VII. Conclusion

The DRWT algorithm enables a fleet of autonomous vehicles to track other vehicles in urban environment in the presence of occlusions. In this method, each sensor-equipped vehicle estimates the target's state over a rolling window, leading to a scalable algorithm that can be parallelized to multiple targets. We show that DRWT converges to the centralized estimate even with less communication bits per node. Future work will focus on target tracking by vehicles with non-linear dynamics and non-linear sensors such as radar and lidar.

References

[1] R. Olfati-Saber, "Distributed Kalman filter with embedded consensus filters," in *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005, pp. 8179–8184.

[2] ——, "Distributed Kalman filtering for sensor networks," in *Decision and Control, 2007 46th IEEE Conference on*. IEEE, 2007, pp. 5492–5498.

[3] ——, "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*. IEEE, 2009, pp. 7036–7042.

[4] G. Battistelli and L. Chisci, "Stability of consensus extended Kalman filter for distributed state estimation," *Automatica*, vol. 68, pp. 169–178, 2016.

[5] Z. Wu, M. Fu, Y. Xu, and R. Lu, "A distributed Kalman filtering algorithm with fast finite-time convergence for sensor networks," *Automatica*, vol. 95, pp. 63–72, 2018.

[6] L.-L. Ong, B. Upcroft, T. Bailey, M. Ridley, S. Sukkarieh, and H. Durrant-Whyte, "A decentralised particle filtering algorithm for multi-target tracking across multiple flight vehicles," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 4539–4544.

[7] A. Ahmad and P. U. Lima, "Multi-robot cooperative object tracking based on particle filters." in *ECMR*. Citeseer, 2011, pp. 37–42.

[8] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri, "Distributed Kalman filtering based on consensus strategies," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 622–633, 2008.

[9] A. W. Stroupe, M. C. Martin, and T. Balch, "Distributed sensor fusion for object position estimation by multi-robot systems," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, vol. 2. IEEE, 2001, pp. 1092–1098.

[10] W. Niehsen, "Information fusion based on fast covariance intersection filtering," in *Proceedings of the Fifth International Conference on Information Fusion. FUSION 2002.(IEEE Cat. No. 02EX5997)*, vol. 2. IEEE, 2002, pp. 901–904.

[11] S. J. Julier and J. K. Uhlmann, "Using covariance intersection for SLAM," *Robotics and Autonomous Systems*, vol. 55, no. 1, pp. 3–20, 2007.

[12] H. Li and F. Nashashibi, "Cooperative multi-vehicle localization using split covariance intersection filter," *IEEE Intelligent Transportation Systems Magazine*, vol. 5, no. 2, pp. 33–44, 2013.

[13] B. Noack, J. Sijs, M. Reinhardt, and U. D. Hanebeck, "Decentralized data fusion with inverse covariance intersection," *Automatica*, vol. 79, pp. 35–41, 2017.

[14] A. Ahmad, G. D. Tipaldi, P. Lima, and W. Burgard, "Cooperative robot localization and target tracking based on least squares minimization," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 5696–5701.

[15] E. D. Nerurkar, S. I. Roumeliotis, and A. Martinelli, "Distributed maximum a posteriori estimation for multi-robot cooperative localization," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 1402–1409.

[16] P. Dames, P. Tokekar, and V. Kumar, "Detecting, localizing, and tracking an unknown number of moving targets using a team of mobile robots," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1540–1553, 2017.

[17] P. Dames, "Distributed multi-target search and tracking using the PHD filter," in *2017 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, 2017, pp. 1–8.

[18] G. Sibley, "Sliding window filters for SLAM," *University of Southern California, Tech. Rep.*, 2006.

[19] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[20] G. Mateos, J. A. Bazerque, and G. B. Giannakis, "Distributed sparse linear regression," *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5262–5276, 2010.

[21] R. T. Rockafellar, "Monotone operators and the proximal point algorithm," *SIAM Journal on Control and Optimization*, vol. 14, no. 5, pp. 877–898, 1976.

[22] E. Montijano, R. Aragues, and C. Sagüés, "Distributed data association in robotic networks with cameras and limited communications," *IEEE Transactions on Robotics*, vol. 29, no. 6, pp. 1408–1423, 2013.

[23] M. A. Manzoor, Y. Morgan, and A. Bais, "Real-time vehicle make and model recognition system," *Machine Learning and Knowledge Extraction*, vol. 1, no. 2, pp. 611–629, 2019.

[24] J.-W. Hsieh, L.-C. Chen, and D.-Y. Chen, "Symmetrical SURF and its applications to vehicle detection and vehicle make and model recognition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 6–20, 2014.

[25] S. Du, M. Ibrahim, M. Shehata, and W. Badawy, "Automatic license plate recognition: A state-of-the-art review," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 2, pp. 311–325, 2012.

[26] T.-H. Chang, M. Hong, and X. Wang, "Multi-agent distributed optimization via inexact consensus ADMM," *IEEE Transactions on Signal Processing*, vol. 63, no. 2, pp. 482–497, 2014.

[27] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.