Traffic Light Status Detection Using Movement Patterns of Vehicles

Joseph Campbell[1], Heni Ben Amor[1], Marcelo H. Ang Jr.[2], and Georgios Fainekos[1]

Abstract— **Vision-based methods for detecting the status of traffic lights used in autonomous vehicles may be unreliable due to occluded views, poor lighting conditions, or a dependence on unavailable high-precision meta-data, which is troublesome in such a safety-critical application. This paper proposes a complementary detection approach based on an entirely new source of information: the movement patterns of other nearby vehicles. This approach is robust to traditional sources of error, and may serve as a viable supplemental detection method. Several different classification models are presented for inferring traffic light status based on these patterns. Their performance is evaluated over real and simulated data sets, resulting in up to 97% accuracy in each set.**

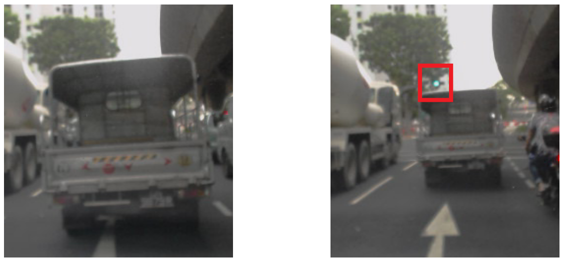
Fig. 1: Example of traffic light occlusion by other vehicles. The light is obscured in the left image but becomes visible in the right image as the traffic starts moving.

I. Introduction

One of the many challenges facing autonomous vehicles is the ability to safely navigate complex environments such as intersections while maintaining compliance with local traffic regulations. Vehicles and pedestrians with varying directions of travel cross paths while being guided by traffic lights that are optimized for identification by human drivers. This issue has been partially addressed with the introduction of intelligent traffic light systems which actively communicate their signal to nearby vehicles through Vehicular Ad Hoc Networks [1], [2]. Nonetheless, as observed by [3], such systems have thus far been limited to small-scale academic experiments and a timely integration into current road networks seems unlikely.

It is for this reason that recent work has focused on the real-time identification of traffic light signals via vision-based systems [3], [4]. This approach can work well but is subject to errors that can lead to the misidentification of the status of traffic light signals. These errors can arise due to poor lighting conditions which interfere with the camera sensor or an obstructed view resulting from a dirty lens or another vehicle, as demonstrated in Fig. 1. This can potentially lead to disastrous results – an autonomous vehicle erroneously passing through an intersection could find itself in a situation in which it is unable to avoid a collision.

This paper proposes a complementary traffic light identification system based on an alternative source of information: the behavior of other nearby vehicles based on positional data. Conceptually, the system infers the status of a traffic light from the movements of other vehicles around or in the corresponding intersection. The advantage of such a system is not that it has fewer failure-inducing cases than a vision-based system, but rather that they are *different* failure cases. Ideally, this system will be paired with a vision-based one such that they complement each other and reduce the total points of failure.

Consider the following scenario: a traffic light is non-functional due to an extenuating circumstance. As is typical on US roads, a law enforcement officer is directing traffic through the intersection. A typical vision-based recognition system is of no help in this scenario, however, by observing when other vehicles start to pass through the intersection and from which direction, the proposed system can infer which traffic the officer is allowing to pass through the intersection.

Similarly, consider a situation in which a vehicle stops at a red light behind a larger vehicle that is occluding the traffic light. Suppose the preceding vehicle suffers a mechanical failure and is blocking traffic; the traffic light cycles through its phases and surrounding traffic bypasses the offending vehicle in other lanes. Once again, a vision-based system would not be of assistance in this scenario; however, the proposed system may indicate that the traffic light is green and, therefore, alternative action should be taken.

The contributions of this paper are as follows: we present a system for predicting the state of a traffic light based on the spatial movement of nearby vehicles, and evaluate its effectiveness in simulated and real-world conditions.

II. Related Work

Vision-based traffic light detection systems have been widely analyzed in previous works. The majority of these works have focused purely on image recognition [5], [6], [7], [8]. Of particular interest, however, are those that seek to minimize the risk posed by errors inherent to vision-based detection systems. In [4], the authors propose using a detailed map of traffic lights to act as prior knowledge so that the

[1]J. Campbell, H. B. Amor, and G. Fainekos are with the School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85281, USA. {jacampb1, hbenamor, fainekos}@asu.edu

[2]M. H. Ang Jr. is with the National University of Singapore, Kent Ridge, Singapore. mpeangh@asu.edu

This material is based upon work supported by the National Science Foundation EAPSI Fellowship under Grant No. 1515589 and CPS 1446730.

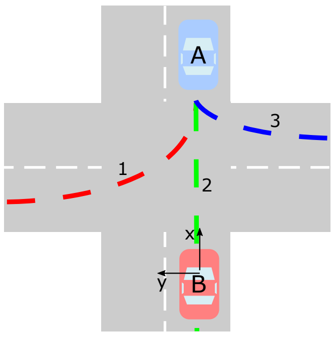

Fig. 2: A scenario in which vehicle A's current position is ambiguous, as there are multiple paths it could have taken which could be used to infer different traffic light states.

detection system knows when it should be able to see traffic lights. If traffic lights are not detected at an expected position, the autonomous vehicle can take preventative action such as slowing down under the assumption that the light is red or yellow. However, this could have unintended consequences since if the light is green this action may result in a collision with human-operated vehicles due to unpredictability.

Similarly, in [3] the authors acknowledge the difficulties in building a purely vision-based traffic light detection system and so augment theirs with prior map knowledge as well as temporal information. While yielding good results, the system still fails to identify traffic light signals in certain cases. Indeed, the authors indicate that a possible approach for improvement would be to introduce 3-dimensional LIDAR data into the mix in order to improve recognition of the traffic lights themselves.

In [9], the authors use the movement patterns of pedestrians to apply semantic labels to the environment. They infer the location of pedestrian crossings, sidewalks, and building entrances and exits based on the activity patterns of pedestrians. This is similar in spirit, if not in execution, to the labeling of traffic lights based on vehicle movement patterns introduced in this paper.

III. Problem Formulation

We can formalize the problem from a probabilistic perspective as follows: let Z be a discrete random variable which represents the state of a traffic light with respect to a target vehicle. The specific value of Z is denoted by z, and in this paper can take the value of either $green$ or red. The goal is to then determine the probability that a traffic light is either $green$ or red with respect to our target vehicle at a specific point in time t: $p(Z_t = z_t)$. To simplify the notation, from this point on we will refer to this probability as simply $p(z_t)$.

Clearly, we cannot determine an accurate probability for $p(z_t)$ without additional information. Therefore, we would like to consider observations of nearby vehicles when determining this probability. The simplest approach is to consider the spatial position of every nearby vehicle independently at each point in time. We define the state g of vehicle n at time t as:

$$g_{n,t} = \left[x_{n,t}, y_{n,t}\right]^T . \tag{1}$$

If we place the target vehicle at the origin of a Cartesian coordinate plane with the positive x-axis extending towards the front of the vehicle and the positive y-axis extending towards the left-hand side of the vehicle, then x is the distance along the x-axis from the target vehicle to the observed vehicle n. Similarly, y is the distance along the y-axis to vehicle n. This yields a conditional probability of the following form, where N is the total number of observable vehicles at time t,

$$p(z_t|g_{1,t}, g_{2,t}, ..., g_{N,t}) = p(z_t|g_{1:N,t}). \tag{2}$$

However, this approach has a potential drawback which is visualized in Fig. 2. If vehicle A is an observable vehicle at time t, it may have taken several different paths to arrive at this position: path 1, 2, or 3. Each of these paths could result in a different traffic light state z_t. For example, if vehicle B is our target vehicle and we are observing A, z_t could be $green$ if vehicle A followed path 2, red if path 1, and either $green$ or red if path 3 (depending on local traffic regulations for right-on-red turns). This leads to an ambiguous situation, in which the state of vehicle A at this point in time does not necessarily help us determine z_t.

We can alleviate this problem if we consider a temporal trace of the position. We could alter the vehicle state to include information on the position over time in the form of velocity,

$$g_{n,t} = \left[x_{n,t}, y_{n,t}, \dot{x}_{n,t}, \dot{y}_{n,t}\right]^T . \tag{3}$$

This is susceptible to the same ambiguity problem, however. In the example from Fig. 2, if path 2 resulted from A accelerating through a light which recently turned $green$, then the velocity at time t could be roughly the same for all paths. The same holds true when acceleration is considered:

$$g_{n,t} = \left[x_{n,t}, y_{n,t}, \dot{x}_{n,t}, \dot{y}_{n,t}, \ddot{x}_{n,t}, \ddot{y}_{n,t}\right]^T . \tag{4}$$

A more effective approach is to consider the state of vehicle n not just for a single time step t, but rather over a time window, i.e., $t-1$, $t-2$, and so on. If we consider a window size of T time steps in the past, then we can represent the state of vehicle n as a time series s at time t:

$$s_{n,t} = g_{n,t}, g_{n,t-1}, ..., g_{n,t-T_n} \tag{5}$$

$$p(z_t|s_{1,t}, s_{2,t}, ..., s_{N,t}) = p(z_t|s_{1:N,t}). \tag{6}$$

If observations are ideal, then the entire path for vehicle A is now taken into account and there is no more ambiguity. In practice, this may not be the case and the effective window size T_n may vary from vehicle to vehicle. For example, A may only enter the sensor range of our target vehicle once it reaches the position depicted in Fig. 2.

Problem: Given a set of observations $\mathcal{L}$ of nearby vehicles, determine $p(z_t|\mathcal{L})$ under the following assumptions.

1) $\mathcal{L}$ is either a set of independent vehicle states g, or a set of independent series of states s.
2) Each series s may contain a variable number of states corresponding to sequential time points.
3) At least one vehicle must be observed for at least one time step.

In practice, Assumption 2 is not strong as this is implicitly satisfied by a Bayesian tracking algorithm in this paper.

IV. Methodology

In order to find the probability in Eq. (6), we adopt a Bayesian interpretation which yields the following posterior:

$$p(z_t|s_{1:N,t}) = \frac{p(s_{1:N,t}|z_t)p(z_t)}{p(s_{1:N,t})}. \tag{7}$$

We employ a discriminative model to find this posterior directly. Artificial neural networks (ANNs) have excellent predictive capabilities given nonlinear input-output mappings, particularly when applying a classification label to a time series input. Depending on the cost function and network complexity, ANNs can also accurately approximate a Bayesian posterior directly [10], in our case $p(z_t|s_{1:N,t})$.

A. Artificial Neural Networks

Artificial neural networks are mathematical models capable of accurately approximating any continuous function [11]. In this work, we employ ANNs in a supervised pattern classification capacity to approximate the posterior probability in Eq. (2). In other words, the input to the network is the feature vector $g_{n,t}$ and the expected output is z_t. Each observed vehicle state $g_{n,t}$ is treated as independent, and the goal is to learn which states correspond to each value of z_t. In a feed-forward neural network (FFNN), the nodes are not allowed to form cycles. This is suitable for simple pattern classification, however, it is not ideal when we would like to consider some inputs as dependent and use multiple inputs to derive a single output. This is the case when approximating the posterior probability in Eq. (6), as it is conditional on a time series of vehicle states $s_{n,t}$ as defined in Eq. (5). This is known as *sequence classification* [12], and recurrent neural networks have been previously used with great effect [13]. A recurrent neural network (RNNs) [14] is a type of neural network that is allowed to form cyclical connections among hidden layer nodes.

We use RNNs to estimate $p(z_t|s_{n,t})$ by generating a single output z_t from a sequence of feature vectors $g_{n,t}$, which together form the time series $s_{n,t}$ of state vectors for vehicle n. However, this is not the same as the posterior probability defined in Eq. (6) which is conditional on all observed vehicles, not just one. The feature vector to our network must be a constant size. This rules out simply concatenating the feature vectors of all observed vehicles, since the number of observed vehicles may vary at any given time. Instead, we take the mean probability of $p(z_t|s_{n,t})$ for all observed vehicles at time t and use that as an approximation:

$$\hat{p}(z_t|s_{1:N,t}) \approx \frac{\sum_{n=1}^{N} \hat{p}(z_t|s_{n,t})}{N}. \tag{8}$$

B. Bidirectional Long Short Term Memory Networks

Standard RNNs suffer from a problem known as the *vanishing gradient* [15], in which the hidden layer node weights for previous inputs converge to zero over time, thus preventing an RNN from effectively learning from inputs that span a long time period. A variant of the RNN known as the Long Short Term Memory (LSTM) [16] network was designed to mitigate this problem by introducing the concept of LSTM nodes that are more effective at retaining previous values. Furthermore, the Bidirectional Long Short Term Memory (BLSTM) network was shown to be exceptionally well-suited for sequence classification [17].

The bidirectional aspect of a BLSTM network is a concept lifted from Bidirectional Recurrent Neural Networks (BRNNs) [18]. In a BRNN, two RNNs – one processing the input series forward in time and one backward in time – are connected to the same output layer. This architecture yields greater prediction accuracy as it predicts based on past inputs as well as future inputs.

V. Experiments

A. Experimental Setup

In order to evaluate how well these networks can approximate the posterior probabilities, we collected two sets of data with which to perform experiments. The first set was generated from real-world sensor data collected by an autonomous vehicle from 56 intersections in the vicinity of the National University of Singapore campus in Singapore. Spatial point cloud data was collected with a SICK LMS 151 LIDAR sensor operating at 50Hz. As there is no ground truth available with which to form the vehicle time series, the data was fed through a two-stage vehicle tracking algorithm.

The first stage decomposes the point cloud data into a subset of clusters, in which each cluster consists of a collection of points in close proximity to each other. The clusters are then tracked over several measurement frames to yield an average spatial position and a velocity vector for a given point in time [19]. The second stage treats these independent measurements as observations to a particle filter-based multi-target tracking algorithm [20]. Vehicle time series are then derived from the particle filters and down-sampled to 10Hz. Supervised labels were manually generated from camera inputs collected simultaneously with the LIDAR data. This process yielded a data set consisting of 1011 unique time series.

Despite the inherent value of real-world data, there is a limit to how much can be collected. Additionally, due to practical constraints, we could only collect data from nearby intersections which limits how well we can generalize. Therefore, we turned to synthetic data generated with the SUMO traffic simulator [21]. Road networks for 13 intersections were generated from OpenStreetMap data: 3 in Tempe, Arizona, 2 in New York City, New York, and 8 in Singapore. Simulations were run in which traffic passed through the intersections from each direction and either traveled straight, turned left, or turned right. Vehicles

Classifier	**Feature Set**	**Real** (No Track)	**Real**	**Sim**	**Sim-Real**	**Sim** (Noisy)	**Sim-Real** (Noisy)
1-NN	x, y	0.678	0.853	0.737	0.719	0.813	0.599
1-NN	$x, y, \dot{x}, \dot{y}$	0.910	0.977	0.968	0.934	838	0.648
1-NN	$x, y, \dot{x}, \dot{y}, \ddot{x}, \ddot{y}$		0.943	0.972	0.969	0.833	0.627
FFNN	x, y	0.669	0.697	0.655	0.690	0.642	0.684
FFNN	$x, y, \dot{x}, \dot{y}$	0.850	0.897	0.796	0.774	0.692	0.716
FFNN	$x, y, \dot{x}, \dot{y}, \ddot{x}, \ddot{y}$		0.899	0.862	0.852	0.695	0.709
BLSTM	x, y		0.655	0.764	0.683	0.765	0.679
BLSTM	$x, y, \dot{x}, \dot{y}$		0.790	0.870	0.740	0.874	0.742
BLSTM	$x, y, \dot{x}, \dot{y}, \ddot{x}, \ddot{y}$		0.782	0.908	0.804	0.863	0.718

TABLE I: The mean test accuracy for 1-Nearest Neighbor (1-NN), Feed-forward Neural Network (FFNN), and Bidirectional Long Short Term Memory (BLSTM) classifiers. The best classifier for each data set is highlighted in green, while the classifiers that are not significantly different are highlighted in yellow.

were uniformly distributed to one of three behavior models: aggressive, average, and submissive.

SUMO is capable of writing floating car data (FCD) output, which contains the position, velocity, and heading of every vehicle at each sampling interval for the duration of the simulation. To correspond with the real data set, the sampling interval was fixed to 10Hz. This data was then transformed with respect to a chosen target vehicle, and used to generate state vectors for each other vehicle within a 50m sensor range of the target. Since the FCD data includes a vehicle identifier, these states can then be assembled into a time series for each vehicle. These time series' were segmented in order to coincide with the states of the intersection's traffic light and labeled as either $green$ or red. This process yielded a data set consisting of 2311 unique time series.

The BLSTM network used in these experiments is composed of an input layer followed by two parallel LSTM layers with 32 nodes each; one layer processes the input sequence forward and one layer backwards. The output from the LSTM layers is concatenated into a dropout layer with a 0.5 drop rate. The FFNN is a standard feed-forward network with 3 hidden layers and 256 nodes per layer. In both networks, the size of the input layer is dependent on the number of vehicle state variables, while the output layer always consists of two nodes in order to produce a one-hot encoding of z_t. The networks are trained using RMSProp backpropagation with categorical cross-entropy loss and a softmax activation function. Additionally, a simple K-Nearest Neighbor algorithm was evaluated to serve as another point of comparison. For all experiments, $K = 1$.

B. Results and Discussion

The first experiment of interest is to evaluate the relative performance of each classifier on our data sets, the results of which are shown in Table I. The classification accuracy is evaluated for the posterior probabilities produced by both the FFNN and BLSTM classifiers, with a train/validation/test set split of 60%/20%/20%, as well as the 1-NN classifier with a 80%/20% train/test split. This experiment reveals that despite being the simplest, the 1-NN classifier performs significantly better than all other classifiers on the *Real* data set with a 97% classification rate. Since this is an unexpected result, we were interested in whether the noise reduction caused by the Bayesian tracking had a significant impact on the 1-NN performance. Thus, we created a *Real (No Track)* data set with only the raw measurements obtained by the clustering algorithm. However, despite slightly reduced performance, the 1-NN classifier is still the best performer on this data set. Results for BLSTM and feature sets containing acceleration are not included for this data set as they require the time series information provided by the tracking algorithm.

Furthermore, 1-NN has the highest classification accuracy on the *Sim*ulation data set. This seems to indicate that the *Sim* data set is a good approximation of the real data set since it yields similar results, but on further analysis the test sample distribution between the two data sets is strikingly different. This can be observed in the first figure of each row in Fig. 3. The *Real* data set is heavily skewed, with a large portion of the test samples coming from intersections where only a small number of vehicles were observed for a short period of time. Meanwhile, the *Sim* data has a much flatter distribution over a wider domain.

In order to determine whether this distribution has a prominent effect on classification accuracy, we ran an optimization algorithm to minimize the Kullback-Leibler divergence between the distributions of the two data sets. This was accomplished by using the truncation factor of a random portion of the time series as the search parameter. The initial KL divergence between the *Real* and *Sim* data sets is 1.35, however, after this optimization routine that was reduced to 0.09. The resulting data set is referred to as *Sim-Real*, and it can be seen in Fig. 3 that the associated test sample distribution is similar to that of the *Real* data set. As in the other data sets, the 1-NN classifier is again the best performer on the *Sim-Real* data and indicates robustness to changes in the test sample distribution. Additionally, since the simulation data yields similar results to the real-world data and is capable of closely approximating the real-world observation distribution, we consider it an accurate representation of the real-world data.

The only time we observed the 1-NN classifier perform poorly is on data sets with a considerable amount of noise. Gaussian noise with a standard deviation of 2.0 was applied to all values in the *Sim* and *Sim-Real* data sets, resulting in *Noisy* variations. The results in Table I show that 1-

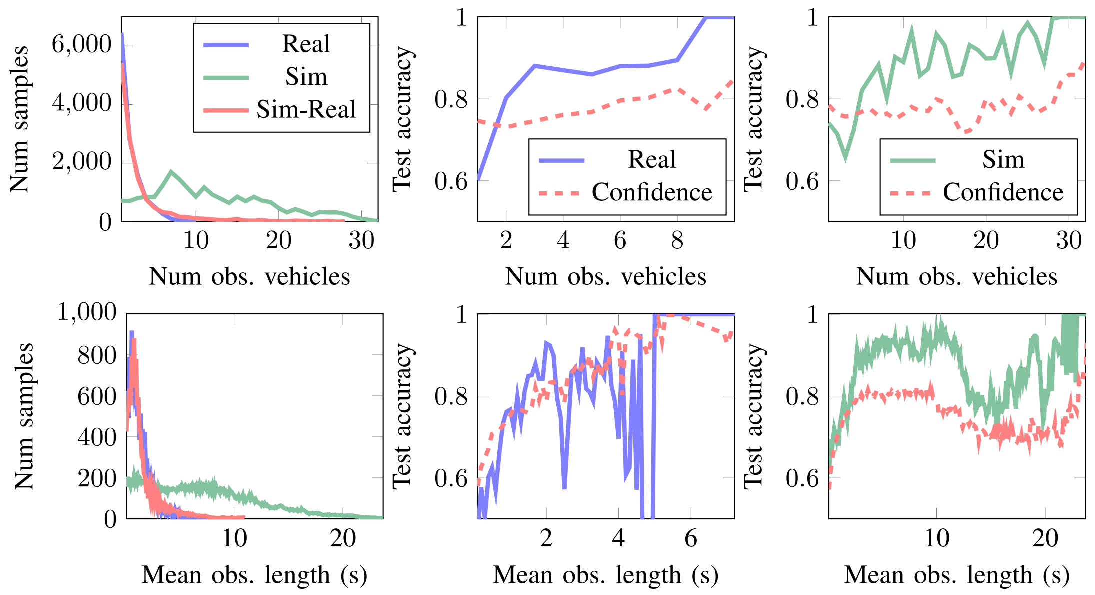

Fig. 3: The first figure in each row shows the distribution of test samples in each data set according to the number of observed vehicles per time step, and the mean observation length among all observed vehicles per time step. The remaining figures in each row show the BLSTM (full feature set) test accuracy for real data (solid blue line) and simulation data (solid green line) at each time step for both criteria. The red dashed indicates the prediction confidence at each time step.

NN yielded a considerably worse classification accuracy in this scenario, while BLSTM was largely unaffected by the additional noise and achieved the best accuracy with 87% and 74% on the *Sim* and *Sim-Real* data sets respectively.

The results in Table I also allow us to examine the impact of the different vehicle states defined in Eqs. (1), (3), and (4) on the overall accuracy. The first observation we can make is that the addition of velocity information into the feature set results in a statistically significant (p-value < 0.05) increase in accuracy for every classifier on every data set. This is a strong result, and in line with the hypothesis that the introduction of velocity information will help alleviate the intersection ambiguity problem. However, it is interesting to note that the addition of acceleration information does not always lead to a further increase in accuracy. The noisy data sets, in particular, actually exhibit either a statistically significant decrease in accuracy or no change at all. This suggests that we can reduce the complexity of our classifiers without penalizing accuracy on noisy data sets by leaving acceleration out of the feature set.

The second experiment is designed to test how the BLSTM classifier would perform in a realistic scenario. In real-world use, we do not have access to the full vehicle time series in the data sets; we only have the vehicle observations that have occurred until the current time step. The network is first trained with the full vehicle time series from all but one of the intersections. With the remaining time series, time is treated as a discrete value and incremented in steps. At every time step, the network is used to estimate the mean temporal probability in Eq. (8) from the vehicle time series that have had an observation within the past 3 seconds. Only

Feature Set	α	**Mean Real**	**Mean Sim**
$x, y, \dot{x}, \dot{y}$	1	0.693	0.861
$x, y, \dot{x}, \dot{y}, \ddot{x}, \ddot{y}$	1	0.718	0.866
$x, y, \dot{x}, \dot{y}$	2	0.837	0.863
$x, y, \dot{x}, \dot{y}, \ddot{x}, \ddot{y}$	2	0.842	0.870
$x, y, \dot{x}, \dot{y}$	3	0.877	0.868
$x, y, \dot{x}, \dot{y}, \ddot{x}, \ddot{y}$	3	0.876	0.876

TABLE II: The BLSTM mean test accuracy for all time steps with at least α observed vehicles.

the observations that have occurred before the current time step are considered. The mean classification accuracy for all time steps is shown in Table II.

With further analysis, it is evident that a significant number of misclassified time steps occur when only one vehicle is observed. As more distinct vehicles are observed, the classification accuracy increases, which is an intuitive result. This is visualized in the top row of Fig. 3. If we examine the distribution of test samples over the number of observed vehicles, it is clear that a large portion of the samples occur when only one vehicle is observable. Taking this into consideration, if the test accuracy is evaluated only for time steps in which two or more vehicles are observed, then the accuracy increases from 71% to 84% on the *Real* data set as shown in Table II. In contrast, the simulation data has a flatter distribution, and as a result the corresponding accuracy does not see a proportional increase. With the positive correlation between accuracy and the number of vehicles, we might also expect such a relationship between the classification accuracy and the length of time that vehicles are observed. The plots in the second row of Fig. 3 show a positive correlation, indicating that this is true to some extent.

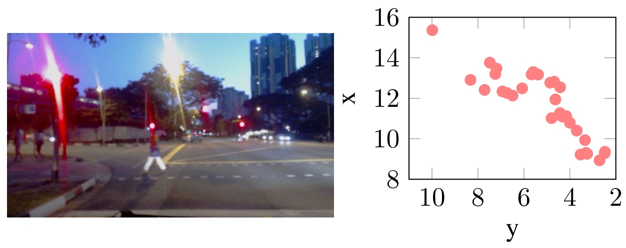

Fig. 4: Scenario in which pedestrians mistaken for a vehicle result in a mis-classification with high confidence. The camera image is on the left, and the corresponding time series given by the particle filter is on the right.

Furthermore, there is also a positive relationship between the accuracy and the BLSTM classifier's prediction confidence, which we define as the maximum probability among all values of z_t. In other words, as the classifier observes more vehicles it grows more confident in the prediction and this results in a higher classification accuracy. However, there are instances in which this does not hold true. Specifically, it can be seen that the accuracy is poor while the prediction confidence is high for the *Real* data set when the mean observation length is between 4s and 5s in Fig. 3. On further analysis, this occurred when the target vehicle was stopped in front of pedestrians crossing a red light as shown in Fig. 4. A single vehicle was tracked for several seconds moving directly in front of the target vehicle with an average speed of $2.83 m/s$. The most likely scenario is that the pedestrians crossing the street were mistaken for a vehicle turning left, which resulted in a prediction of a green light when in fact, the light was red.

VI. Conclusion

This paper has shown that it is possible to accurately infer the current state of a traffic light by analyzing the spatial movements of nearby vehicles with respect to a target vehicle. This method was evaluated on real-world data gathered in Singapore and synthetic data generated from a traffic simulator. In both cases, encouraging results were achieved with three different classifiers: a feed-forward neural network, a bidirectional long short-term memory network, and a nearest neighbor classifier. It was found that in most tested scenarios, a nearest neighbor classifier obtained the best classification results. However, if the data is particularly noisy, better accuracy may be achieved with a BLSTM classifier.

Similar to a vision-based approach, the methodology presented here has failure cases in which inference produces wrong results. The most obvious case is when no vehicles are in observation range, however, it was also seen that in some scenarios more than one vehicle may need to be in observation range in order to make an accurate prediction. Likewise, there are specific instances in which the inference may be wrong if other vehicles are only observed for an extremely brief period of time. However, since the failure cases for our approach and a vision-based approach are not the same, we envision that the best use of our system is to combine it with a traditional vision-based method. Different failure cases suggests that the systems will complement each other, and result in a more robust detection system.

References

[1] N. Maslekar, M. Boussedjra, J. Mouzna, and H. Labiod, "Vanet based adaptive traffic signal control," in *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*. IEEE, 2011, pp. 1–5.

[2] V. Gradinescu, C. Gorgorin, R. Diaconescu, V. Cristea, and L. Iftode, "Adaptive traffic lights using car-to-car communication," in *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*. IEEE, 2007, pp. 21–25.

[3] J. Levinson, J. Askeland, J. Dolson, and S. Thrun, "Traffic light mapping, localization, and state detection for autonomous vehicles," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5784–5791.

[4] N. Fairfield and C. Urmson, "Traffic light mapping and detection," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5421–5426.

[5] J.-H. Park and C.-s. Jeong, "Real-time signal light detection," in *2008 Second International Conference on Future Generation Communication and Networking Symposia*. IEEE, 2008, pp. 139–142.

[6] H. Tae-Hyun, J. In-Hak, and C. Seong-Ik, "Detection of traffic lights for vision-based car navigation system," in *Advances in Image and Video Technology*. Springer, 2006, pp. 682–691.

[7] F. Lindner, U. Kressel, and S. Kaelberer, "Robust recognition of traffic signals," in *Intelligent Vehicles Symposium, 2004 IEEE*. IEEE, 2004, pp. 49–53.

[8] R. De Charette and F. Nashashibi, "Real time visual traffic lights recognition based on spot light detection and adaptive traffic lights templates," in *Intelligent Vehicles Symposium, 2009 IEEE*. IEEE, 2009, pp. 358–363.

[9] B. Qin, Z. J. Chong, T. Bandyopadhyay, M. H. Ang, E. Frazzoli, and D. Rus, "Learning pedestrian activities for semantic mapping," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6062–6069.

[10] M. D. Richard and R. P. Lippmann, "Neural network classifiers estimate bayesian a posteriori probabilities," *Neural computation*, vol. 3, no. 4, pp. 461–483, 1991.

[11] K.-I. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural networks*, vol. 2, no. 3, pp. 183–192, 1989.

[12] Z. Xing, J. Pei, and E. Keogh, "A brief survey on sequence classification," *SIGKDD Explor. Newsl.*, vol. 12, no. 1, pp. 40–48, Nov. 2010.

[13] A. Graves, "Sequence transduction with recurrent neural networks," *ICML Representation Learning Worksop*, 2012.

[14] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[15] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[18] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *Signal Processing, IEEE Transactions on*, vol. 45, no. 11, pp. 2673–2681, 1997.

[19] X. Shen, S.-W. Kim, and M. Ang, "Spatio-temporal motion features for laser-based moving objects detection and tracking," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 4253–4259.

[20] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers, "Tracking multiple moving targets with a mobile robot using particle filters and statistical data association," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 2. IEEE, 2001, pp. 1665–1670.

[21] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO - Simulation of Urban MObility," *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, pp. 128–138, December 2012.