Factor Graph Scene Distributions for Automotive Safety Analysis

Tim A. Wheeler and Mykel J. Kochenderfer

***Abstract*— Automotive safety validation requires evaluation on a statistically representative set of roadway configurations and scene geometries. Scenes must be sampled from a statistical model representative of what actually occurs on roadways. This paper introduces a methodology for realistic scene model construction based on factor graphs that can be applied to arbitrary road geometries. Parameter learning for factor graphs is known to be convex. Experiments show that the proposed method is superior to the state of the art.**

I. Introduction

Rigorous analysis of advanced driver assistance systems is required before public release due to the potentially catastrophic consequences of error in their operation. Transportation authorities, such as the National Highway Traffic Safety Administration (NHTSA) and the Federal Motor Transport Authority, require a combination of drive tests and detailed simulation studies to ensure system effectiveness and safety [1]. A drive test can evaluate a system in actual operation, but simulation studies are required to test the robustness of the system over a significantly wider range of situations. These situations, in turn, need to be generated by a statistical model of scenes that are representative of what actually occurs on the roadway. Sampling a large collection of scenes from such a model and running them in simulation both with and without a driver assistance system provides an estimate of the differential in performance and safety.

Sampling scenes by simulating traffic over a burn-in period does not guarantee appropriate roadway population and vehicle characteristics. It is important that the scene geometries and inter-vehicle relations represented by the model be as representative of actual driving as possible; otherwise, the risk associated with an advanced driver assistance system could be significantly over- or underestimated. To ensure a representative model, a large collection of recorded driving data is typically used to extract probabilities over various encounter variables. This statistical method of safety system evaluation has been used successfully in other fields, including civil aviation [2], [3].

The automotive industry has traditionally tested on a restricted set of pre-selected scenarios or has sampled directly from or directly replayed recorded driving data [4], [5] A statistical method was recently introduced for learning automotive scene models from data [6]. This model used a Bayesian network to represent relationships between vehicles and sampled from conditional distributions to generate scenes. Models were constrained to a particular roadway geometry and did not consider interactions between lanes.

Tim A. Wheeler and Mykel J. Kochenderfer are with Aeronautics and Astronautics, Stanford University, Stanford, CA 94305, USA {wheelert, mykel}@stanford.edu

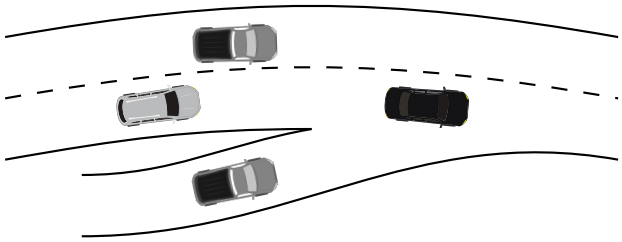


Fig. 1: An example scene on an arbitrary road network.

The scene models described in this paper extend these prior methods in several important ways. First, these models allow for arbitrary roadway configurations. Second, these models produce scenes that populate the roadway as it is used in the real world. Third, use of a factor graph allows for the seamless addition of variables to extend the model and capture additional dependencies, and can be trained within a principled statistical framework. Finally, the proposed method is convex and training converges to the global optimum.

II. Scene Distribution as a Factor Graph

A scene s is an arrangement of vehicles on a given road network. The ith vehicle is defined with a length, a width, and a four-tuple $\vartheta^{(i)} = \langle x, y, v, \psi \rangle$ containing two position coordinates, speed, and a lane-relative heading. Vehicles are not constrained to lane centerlines.

A scene distribution $p(s)$ is a probability distribution over driving scenes. It is used to generate initial conditions from which simulation evaluation can be conducted. By running a large collection of scenarios sampled from the scene distribution, one can measure the effectiveness of an advanced driver assistance system.

A scene distribution model should allow for efficient sampling of driving scenes representative of the real world. Varying roadway compositions and traffic participant quantities make it difficult to directly model such a probability distribution.

This work uses a factor graph to represent the scene distribution. A factor graph is a bipartite graph representing the factorization of an unweighted probability distribution [7]. A factor ϕ over a set of random variables $x_{1:N}$ is a mapping of those variables to a non-negative real value. The value associated with a particular variable assignment denotes the affinity to that assignment, with larger values indicating

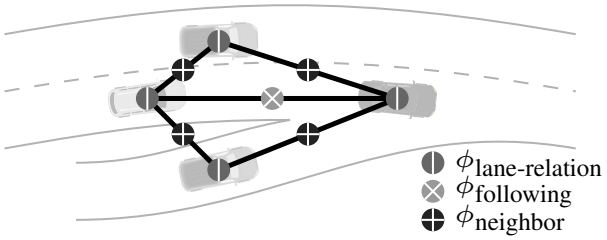


Fig. 2: An example scene graph. Circles represent factors over vehicle parameters. The factor graph is the bipartite graph between the variables defining each vehicle and the feature functions contained in each factor.

higher likelihood. A factor graph is evaluated by taking the product over all factors and is normalized to obtain a probability distribution:

$$p(x_{1:N}) = \frac{1}{Z}\tilde{p}(x_{1:N}) = \frac{1}{Z}\prod_i \phi_i(x_{\text{Scope}[\phi_i]}), \tag{1}$$

where $\tilde{p}$ is the unnormalized measure, Z the normalization constant, and $\text{Scope}[\phi]$ the variables included in ϕ.

Factor graphs allow for an intuitive description of probabilistic interaction between variables. Individual factors can be tuned according to domain knowledge. The normalization constant, however, is often difficult or intractable to compute, and requires exponential time in the number of variables [7].

Log-linear models are one common generalization of parameterized factor graphs [7]. Given a set of features $\mathcal{F} = \{f_i(x_{1:N})\}_{i=1}^K$, where $f_i(x_{1:N})$ is a feature function defined over the factor graph variables, one has:

$$p(x_{1:N} \mid \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})}\exp\left[\sum_{i=1}^K \theta_i f_i(x_{1:N})\right]. \tag{2}$$

The resulting representation is generic and can capture distributions with global and local structure. Factor graphs are useful for modeling distributions over scenes as they can capture probabilistic interactions using a representative set of factors. These factors represent interactions in vehicle state, local context, lane relations, and interactions between vehicles. A factor graph can be constructed for any scene using representative factors in a way that correctly reflects the roadway and vehicle configuration. The feature weights $\boldsymbol{\theta}$ have an intuitive effect on the distribution and can be fit to real-world data.

Figure 2 shows a small example scene factor graph. We include three types of factors: a *lane-relation* factor between a vehicle and the road, which captures the effect of lane-relative features on the vehicle's state; a *following* factor between a vehicle and its lead, which captures information such as time headway; and a *neighbor* factor, which links two neighboring vehicles between lanes. The factor graph approach allows for arbitrary factors to be used; other factors or variables can be included as needed.

A. Factor Graph Construction

To construct scene factor graphs, leading and trailing vehicles were first identified for each vehicle in the scene. Vehicles with both were considered active, and could vary in subsequent scene sampling. Vehicles without leading or training vehicles were held static in order to provide bounds on longitudinal position for the active vehicles. A lane-relation factor was assigned to each active vehicle and a following factor was assigned to each lead-follow vehicle pair. For each active vehicle, a neighbor factor was assigned to the longitudinally closest vehicle in each neighboring lane, both ahead and behind, up to a 33 m horizon.

Each factor follows the log-linear model, containing a set of feature functions. The lane-relation factor is defined over speed v, lane centerline lateral offset t, and lane-relative heading ϕ. The following factor is defined over the relative speed r and headway distance d. All variables are standardized using the mean and variance across the training data. The lane-relation and following factors are each polynomials over their feature variables, with each possible combination of intermediates up to degree 3. For example,

$$\mathcal{F}_{\text{following}} = \left\{r, d, r^2, d^2, r^3, d^3, rd, r^2d, rd^2\right\} \tag{3}$$

The neighbor factor is defined using the time of closest point of approach $t_{\text{CPA}}[\text{s}]$ and distance at closest point of approach $d_{\text{CPA}}[\text{m}]$ assuming each vehicle maintains constant heading and velocity, and takes into account each vehicle's bounding box. Five indicator functions are included:

$$\mathcal{F}_{\text{neighbor}} = \begin{cases} 1\{ \quad t_{\text{CPA}} = 0 \quad \wedge \; d_{\text{CPA}} = 0 \quad \} \\ 1\{ \; 0 < t_{\text{CPA}} \leq 1 \quad \wedge \; d_{\text{CPA}} \leq 1/2 \; \} \\ 1\{ \; 1 < t_{\text{CPA}} \leq 4 \quad \wedge \; d_{\text{CPA}} \leq 1/2 \; \} \\ 1\{ \; 4 < t_{\text{CPA}} \leq 10 \; \wedge \; d_{\text{CPA}} \leq 1/2 \; \} \\ 1\{ 10 < t_{\text{CPA}} \quad \wedge \; d_{\text{CPA}} > 1/2 \; \} \end{cases} \tag{4}$$

B. Scene Sampling

The Metropolis-Hastings algorithm can draw samples from any probability distribution provided that probability density can be evaluated up to a normalization constant [8]. A transition distribution, $\mathcal{T}(x \to x')$, which defines the probability of transitioning from the present state to a successor state, is used to feed a weighted random walk over the problem domain. At every step of the walk, a proposal transition $x \to x'$ is sampled from the transition distribution, and the transition is accepted with probability:

$$\mathcal{A}(x \to x') = \min\left[1, \frac{\tilde{p}(x')\mathcal{T}(x' \to x)}{\tilde{p}(x)\mathcal{T}(x \to x')}\right]. \tag{5}$$

This forms a Markov chain whose stationary distribution is asymptotically equal to the distribution defined by the Markov network [7].

The Metropolis-Hastings algorithm has the advantage of working on unnormalized distributions, and so can use the Markov network factors directly without computing the normalization constant. A disadvantage is that the number of burn-in samples required before convergence to the stationary distribution may be large. Algorithm 1 shows the scene sampling algorithm.

Algorithm 1 Scene Generation using Metropolis-Hastings

Given: transition distribution $\mathcal{T}$
Sample scene s uniformly from a database of real-world scenes
for some number of burn-in steps **do**
    Select an active vehicle $\vartheta_{\text{current}}$ in s uniformly at random
    $\vartheta_{\text{propose}} \leftarrow \vartheta_{\text{current}} + \text{SAMPLE}(\mathcal{T})$
    if $\text{SAMPLE}(U[0,1]) < \mathcal{A}(\vartheta_{\text{current}} \rightarrow \vartheta_{\text{propose}})$ **then**
        $\vartheta \leftarrow \vartheta_{\text{propose}}$
return s

III. Parameter Learning

Parameters for the Markov network factors are learned from real-world data using gradient ascent on the pseudolikelihood, an alternative optimization objective to the likelihood that is commonly used in parameter learning for log-linear models [7]. The likelihood of an assignment is given by $p(x_{1:N}) = \prod_j p(x_j \mid x_{1:j-1})$, which is closely approximated by the pseudolikelihood, $PL(x_{1:N}) = \prod_j p(x_j \mid x_{-j})$, where x_{-j} are all variables except x_j. The log-pseudolikelihood of a dataset with M samples is:

$$\ell_{PL}(\theta \mid \mathcal{D}) = \frac{1}{M}\left(\sum_m \sum_j \ln \tilde{p}(x_j^{(m)} \mid x_{-j}^{(m)})\right) - M \ln Z_{PL}(\boldsymbol{\theta}). \quad (6)$$

It can be shown that the log-pseudolikelihood partition function $\ln Z_{PL}(\boldsymbol{\theta})$ is convex [7], which ensures that gradient ascent converges to the global optimum.

The pseudolikelihood's repeated evaluation of $\tilde{p}(x_j \mid x_{-j})$ only requires integrating out a single variable:

$$\tilde{p}(x_j \mid x_{-j}) = \frac{\tilde{p}(x_j, x_{-j})}{\int_{x_j'} \tilde{p}(x_j', x_{-j})\, dx_j'}. \quad (7)$$

We estimate the denominator with Monte Carlo integration [9]:

$$\int_{x_j'} \tilde{p}(x_j', x_{-j})\, dx_j' \approx \frac{1}{A} V \sum_i \tilde{p}(x_j^{(i)}, x_{-j}), \quad (8)$$

where $V = \int_{x_j'} dx_j'$ and A samples of x_j are uniformly sampled within the variable's domain.

The pseudolikelihood gradient is given by:

$$\nabla_{\theta_i} \ell_{PL}(\theta \mid \mathcal{D}) = \sum_{j : x_j \in \text{Scope}[f_i]} \left(\frac{1}{M}\sum_{m=1}^{M} f_i\left(x_j^{(m)}, x_{-j}^{(m)}\right) - \mathbb{E}_{x_j \sim \tilde{p}_\theta\left(\cdot \mid x_{-j}^{(m)}\right)}\left[f_i\left(x_j, x_{-j}^{(m)}\right)\right]\right). \quad (9)$$

The gradient is straightforward to compute, as each expectation term merely requires a summation over a single random variable conditioned on all of its neighbors. We use importance sampling [10] with a uniform distribution to compute this efficiently:

$$\mathbb{E}_{x_j \sim \tilde{p}_\theta(\cdot \mid x_{-j})}[f(x_j, x_{-j})] \approx \frac{\sum_r f(x_j^{(r)}, x_{-j}) \cdot W_r}{\sum_r W_r}, \quad (10)$$

where $W_r = \tilde{p}_\theta(x_j^{(r)}, x_{-j}) / U(x_j^{(r)})$ is the importance sampling weight, R samples of x_j are uniformly sampled within the variable's domain, and $U(x_j^{(r)})$ is the probability density for the uniform proposal distribution.

A. Variable Bounds

We must bound our variables in order to perform Monte Carlo integration and importance sampling. Recall that vehicles are defined according to $\vartheta = \langle x, y, \theta, v \rangle$ with position, heading, and speed. Speed is bounded by the observed extrema.

To bound the other variables, we change from the global frame to a Frenet-Serret frame [11], where the lane offset is bounded to the lane and the heading is bounded based on the observed lane-relative headings. Thus only the location along the lane remains unbounded.

This work enforces a bound on location along the lane through the preceding and following vehicle. A vehicle's longitudinal position can only vary between the available space it has on the road, imposed by the scene structure.

B. Varying Network Structure

In traditional undirected graphical model learning, the network is assumed constant. In this problem, the scene structure will vary with the scene. This variability complicates learning because the pseudolikelihood must be maximized over different network topologies.

The pseudolikelihood for shared scene factors on a dataset of M scenes is given by:

$$\ell_{PL}(\theta \mid D) = \frac{1}{M}\sum_{m=1}^{M} \sum_{\vartheta \in s^{(m)}} \sum_{x_j \in \left\{\vartheta_s, \vartheta_t, \vartheta_v, \vartheta_\phi\right\}} \ln \tilde{p}(x_j \mid x_{-j}) \quad (11)$$

The pseudolikelihood gradient is given by:

$$\nabla_{\theta_{it}} \ell_{PL}(\theta \mid D) = \sum_{j : x_t \in \text{Scope}[f_{it}]} \left[\frac{1}{M}\sum_{m=1}^{M}\left(\sum_{\phi_t \in s^{(m)}} f_{it}\left(x_j^{(m)}, x_{-j}^{(m)}\right) - \mathbb{E}_{x_j \sim \tilde{p}_\theta\left(\cdot \mid x_{-j}^{(m)}\right)}\left[f_{it}\left(x_j, x_{-j}^{(m)}\right)\right]\right)\right] \quad (12)$$

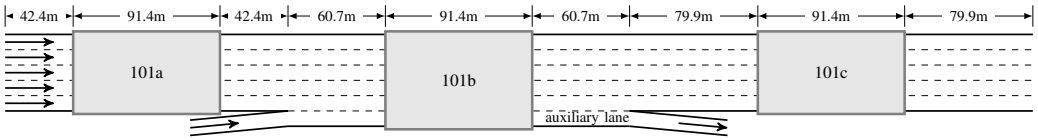
where f_{it} and θ_{it} are the ith feature and weight of the tth shared factor, respectively.

C. Regularization

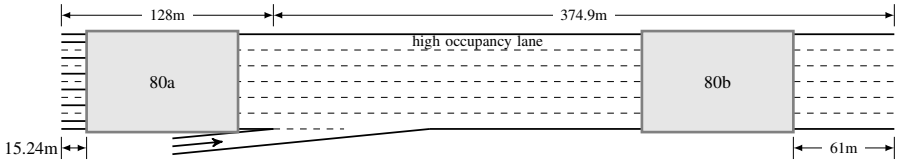
Despite a convex problem formulation, multiple solutions can exist, such as equivalent solutions under parameter scaling [7]. Regularization was employed to ensure well-behaved convergence using a Gaussian prior over the log-linear parameters $\boldsymbol{\theta}$. This regularization preserves convexity.

D. Batch Gradient Ascent

Vanilla gradient descent computes the gradient at each step using all of the scenes in the training set. Faster training time was achieved using batch gradient ascent with momentum [12], which uses randomly sampled batches from the full dataset to speed up gradient computation time, while almost surely converging to the correct answer.



(a) Study Area for Highway 101



(b) Study Area for Interstate 80

Fig. 3: Data source schematics with scene sections. Each scene section is 91.4 m long.

IV. Dataset

This work used real-world driving data obtained from the Next-Generation Simulation (NGSIM) US Highway 101 and Interstate 80 datasets [13], [14]. Each dataset consists of 45 minutes of vehicle trajectory data collected using synchronized digital video cameras providing the vehicle lane positions and velocities at 10 Hz. The US Highway 101 dataset covers an area in Los Angeles, CA, approximately 640 m in length with five mainline lanes and a sixth auxiliary lane providing highway entrance and exit. The Interstate 80 dataset covers an area in the San Francisco Bay Area approximately 500 m in length with six mainline lanes, including a high-occupancy vehicle lane and an onramp. Figure 3 shows the roadways and the scene locations used in our experiments. These datasets were collected by the Next-Generation Simulation program in 2005 to facilitate automotive research and are freely available.

Traffic density in the datasets transitions from uncongested to full congestion and exhibits a high degree of vehicle interaction as vehicles merge on and off the highway and must navigate in the nearly-congested flow. This diversity and the datasets' complete scene description make these sources particularly useful for learning traffic scene distributions.

The NGSIM datasets provide positions and velocities. The trajectories were smoothed using an extended Kalman filter [15] on a simple bicycle model and projected to lanes using centerlines extracted from the NGSIM CAD files.

V. Experiments

The experiments presented in this section use highway scene models trained on the NGSIM dataset. The factor graph model is compared to the 'chain' Bayesian network model from prior work [6], which generates scenes from scratch by using conditional distributions to successively generate a lane of vehicles, one lane at a time. Though identical scene regions are used in order to facilitate a direct comparison, the factor graph model is not restricted to straight highway segments.

A. Burn-in Tuning

An important step in model validation is verifying the implementation of the sampling scheme. To verify the sampling scheme, we generated a large collection of scenes from which feature distributions can be compiled and compared to the observed feature distributions. We began by verifying that the marginal distributions over factor features closely match by comparing their histograms, and used this to set the burn-in time for Metropolis-Hastings to 1000 steps.

Figure 4 shows the real-world scene, the scene factor graph, and a corresponding sampled scene. Note that the inactive vehicles and the fore and rear of the scene do not change during sampling, and that the resulting scene adheres to the same structure. Most importantly, sampled scenes qualitatively look real.

B. Emergent Metrics

Comparing variables not explicitly included in the model to those extracted from a sampled dataset is one method for ensuring that a generative model is representative of the real world. A selection of marginal and emergent metrics are shown in Fig. 5. These were extracted from a set of 10 000 scenes sampled from both the prior Bayesian network model and the new factor graph model. The Kullback-Leibler divergence [16] over a uniform bin-width piecewise-uniform discretization for each metric is included as a bar chart. A smaller bar indicates a closer match to the true distribution.

Results show that the factor graph model produces a closer match to the true distribution over all candidate metrics. The factor graph model clearly outperforms in speed, lane heading, and headway. The factor graph model produces far fewer scenes with unrealistically small timegaps and headways. Both models perform nearly equally well in lane offset, perhaps due to it being largely independent of the lead and trailing vehicle.

The factor graph and Bayesian network models differ in several ways. First and foremost, the Bayesian network model generates scenes from scratch, one lane at a time, by generating a chain of vehicles using a series of conditional probability samples. The factor graph model uses the Metropolis-Hastings algorithm to adjust an existing scene to obtain a new one with the same network structure. This ensures that the roadway remains populated in the same manner as it was in the original scene. Populated lanes remain populated, and empty lanes remain empty.

Fig. 4: The leftmost scene is from the NGSIM dataset, the center scene includes the factor graph structure, and the rightmost scene was sampled from this factor graph using Metropolis-Hastings with 1000 burn-in steps.

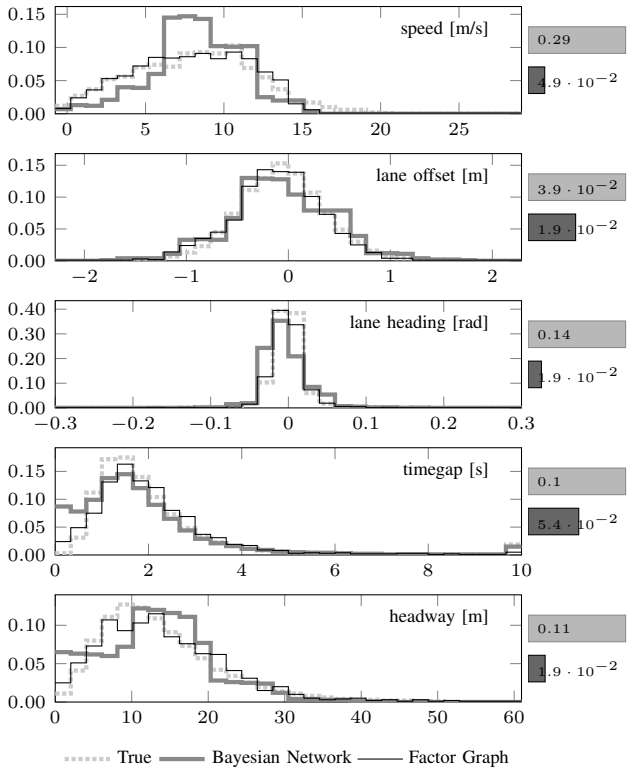


Fig. 5: Extracted metrics from scene datasets, comparing the previous Bayesian network model to the new factor graph approach. Line plots indicate the probability distribution over each metric and the bar plots indicate the Kullback-Leibler divergence between the model distributions and the real world distribution. Smaller KL-divergence values indicate a closer match.

Secondly, the factor graph model includes relationships across lanes and enforces relationships between multiple vehicles at a time. The Bayesian network does not include relationships across lanes and only generates vehicles according to a leader / follower relationship.

Finally, the factor graph model is based on continuous features, whereas the Bayesian network model was originally based on discrete features. While neither model is constrained to either continuous or discrete variables, inference and structure learning in Bayesian networks is made easier with the use of discrete variables, but leads to a trade-off between resolution and the curse of dimensionality. We tried discrete variables with the factor graph and this also outperformed the Bayesian network model.

C. General Road Networks

In this section we investigate the performance of the factor graph approach on larger scenes and general road networks. Scenes were extracted at 1 Hz from the completely populated Highway 101 roadway, including both the mainline lanes and the auxiliary lane. These scenes are approximately 600 m in length. An example scene in shown in Fig. 6. A factor graph model was trained on the dataset of full scenes and a dataset of 1000 scenes was sampled with 1000 burn-in steps.

Metrics were extracted separately for vehicles on the Highway 101 mainline lanes and the auxiliary lane. These are shown in Fig. 7. Vehicles in the mainline lanes account for 98 % of all vehicles.

The factor graph model shows close agreement with the true dataset across all metrics for both mainline and auxiliary vehicles. The auxiliary lane exhibits more positive lane centerline offsets and headings due to the frequent lane changes from vehicles merging onto the highway. The headway distribution for the auxiliary lane is more uniform, due to infrequent vehicles. Vehicles in the auxiliary lane also tend to drive faster than in the congested mainline lanes. Nevertheless, because of the conditional coupling learned through the following factors, the factor graph model still produces valid vehicles in the auxiliary lane.

VI. Conclusions

This paper developed a factor graph model for scene generation on arbitrary highway topologies that closely matches the true distribution of driving scenes. This paper also showed how to sample from the factor graph model using the Metropolis-Hastings algorithm, which allows for sampling initial scenes from real-world scenes, ensuring that highways are realistically populated. Empirical demonstrations show that the proposed method is superior to the state of the art in producing sample distributions which reflect real-world scenes.

Future work will investigate alternative factor features, alternative methods for the proposal distribution, and include additional information in the dataset, such as brake light indicators. The method's performance should be investigated on more complicated road networks, such as merging, roundabouts, and toll booths, and should be compared to scene generation via simulation burn-in. All software is publicly available at `github.com/sisl/2016_itsc_scenegen`.

Acknowledgment

This work was sponsored by Robert Bosch LLC. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by Bosch. The authors appreciate the support and assistance provided by the Bosch automated driving team and gratefully acknowledge the helpful comments from anonymous reviewers.

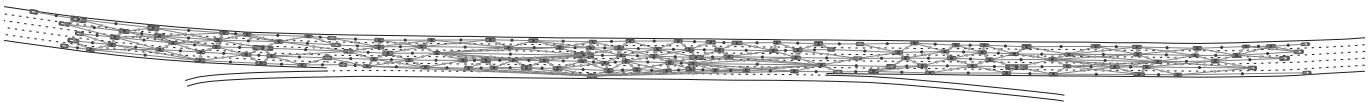


Fig. 6: A factor graph over a full-size scene on Highway 101.

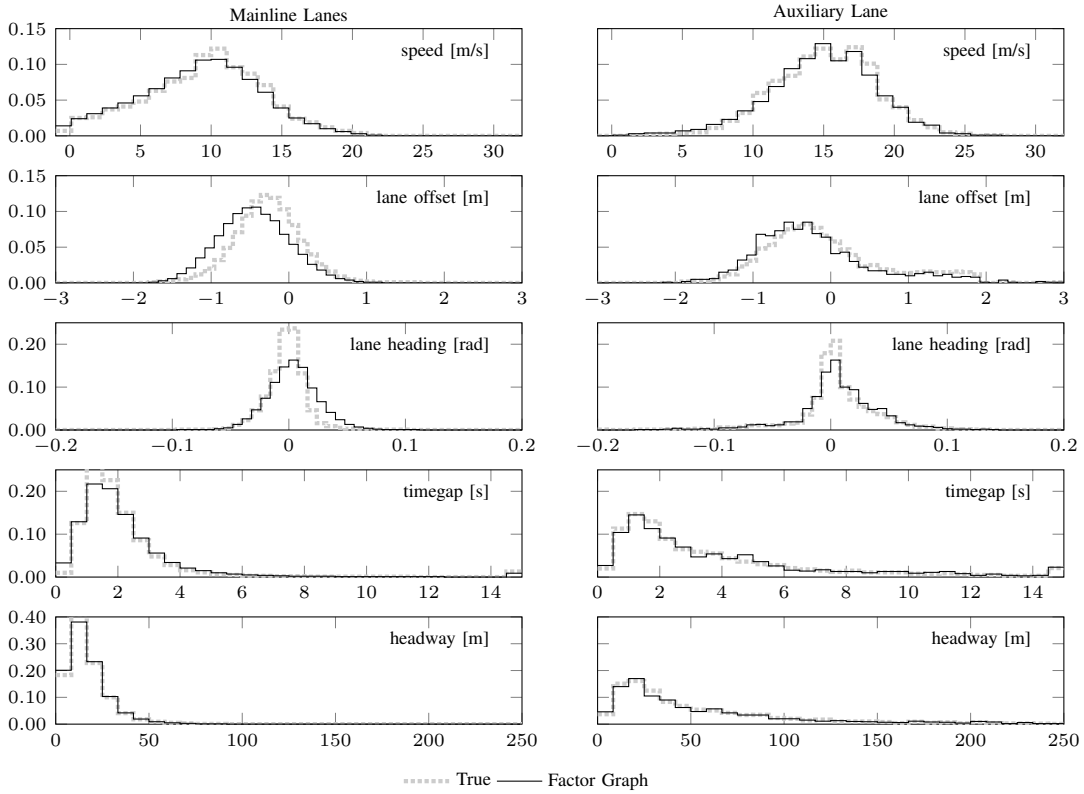


Fig. 7: Scene metric distributions for scenes covering the full Highway 101 road network.

REFERENCES

[1] "ISO 26262-1:2011(en) road vehicles - functional safety," International Standardization Organization, Tech. Rep., 2011.

[2] M. J. Kochenderfer, M. W. Edwards, L. P. Espindle, J. K. Kuchar, and D. J. Griffith, "Airspace encounter models for estimating collision risk," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 33, no. 2, pp. 487–499, 2010.

[3] M. J. Kochenderfer, J. E. Holland, and J. P. Chryssanthacopoulos, "Next-generation airborne collision avoidance system," *Lincoln Laboratory Journal*, vol. 19, no. 1, pp. 17–33, 2012.

[4] "Google self-driving car testing report on disengagements of autonomous mode," Google Auto, LLC, Mountain View, CA, Tech. Rep., Dec. 2015.

[5] A. Zlocki, L. Eckstein, and F. Fahrenkrog, "Evaluation and sign-off methodology for automated vehicle systems based on relevant driving situations," *Journal of the Transportation Research Board*, no. 2489, pp. 123–129, 2015.

[6] T. A. Wheeler, P. Robbel, and M. J. Kochenderfer, "Initial scene configurations for highway traffic propagation," in *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2015.

[7] D. Koller and N. Friedman, *Probabilistic graphical models: Principles and techniques*. Cambridge, Massachusetts: MIT Press, 2009.

[8] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[9] R. E. Caflisch, "Monte Carlo and quasi-Monte Carlo methods," *Acta Numerica*, vol. 7, pp. 1–49, 1998.

[10] R. Srinivasan, *Importance sampling: Applications in communications and detection*. Springer, 2002.

[11] F. Frenet, "Sur les courbes à double courbure," *Journal des mathématiques pures et appliquées*, vol. 17, pp. 437–447, 1852.

[12] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[13] J. Colyar and J. Halkias, "US highway 101 dataset," Federal Highway Administration (FHWA), Tech. Rep. FHWA-HRT-07-030, Jan. 2007.

[14] J. Colyar and J. Halkias, "US highway 80 dataset," Federal Highway Administration (FHWA), Tech. Rep. FHWA-HRT-06-137, Dec. 2006.

[15] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT Press, 2005.

[16] S. Kullback and R. A. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, pp. 79–86, 1951.