Decision-Theoretic Reasoning for Traffic Monitoring and Vehicle Control

Michael P. Wellman, Chao-Lin Liu, David Pynadath
Department of Electrical Engineering and Computer Science, University of Michigan

Stuart Russell, Jeffrey Forbes, Timothy Huang, Keiji Kanazawa*
Computer Science Division, University of California Berkeley

Abstract

We describe technology for robust traffic monitoring and automated vehicle control using decision-theoretic and probabilistic reasoning methods. In this work, we have designed and implemented probabilistic models for deriving high-level descriptions of traffic conditions, as well as the maneuvers and intentions of individual vehicles, from visual observation of a traffic scene. Enhancements to standard probabilistic modeling and inference techniques have improved the performance of uncertain reasoning over time with continuous variables. We have demonstrated our models and algorithms in real-time analysis of traffic images as well as control of simulated vehicles.

1 Overview

This paper describes models and algorithms for deriving high-level descriptions of traffic conditions—as well as the maneuvers and intentions of individual vehicles—from visual or sensor-based observation of a traffic scene. Our probabilistic network models represent both individual vehicles (their position, velocity, etc.) and aggregate variables concerned with the interaction of vehicles (flow, travel time, etc.). This high-level reasoning has been successfully integrated with real-time visual processing to identify and track individual vehicles accurately. In addition to modeling the traffic situation, we can use the probabilistic networks to model the sensors as well, demonstrating integrated sensor fusion and validation.

This framework has been extended to handle reasoning over time, by adding nodes to represent state variables at progressive time slices. At the next level of abstraction, our plan recognition framework can capture the beliefs and intentions of individual drivers and represent the dependence of their actions. This dependence encapsulates the driver's decision-making process in choosing a maneuver, based on goals and knowledge. With the resulting probabilistic model, we can use partial observations (e.g., lane changes, signals) of a vehicle to infer the driver's plan (e.g., passing, exiting) and future actions.

In addition to traffic monitoring on stationary highway video, we have demonstrated the use of similar integrated sensing and probabilistic reasoning in dynamic vehicle traffic simulation. Through integration with a simple decision-tree based decision-making system, we have demonstrated autonomous intelligent driving. Our driving system is able to maintain a sensible estimate of the current traffic situation from simulated sensor inputs and negotiate a variety of challenging traffic contingencies. Our simulator is integrated with the SmartPATH animation system for real-time visualization of the traffic scenarios.

Complex probabilistic networks often require prohibitive computational resources for practical real-time traffic monitoring. We have designed, implemented, and demonstrated new approximation algorithms for probabilistic network inference especially well-suited to continual state updating and predicting, as often required for traffic monitoring and control. These algorithms aim to produce the most accurate predictions possible within the time available for inference.

2 Bayesian Networks

Recent advances in probabilistic modeling technology from the AI/uncertain reasoning community have led to significant improvements in the representation of uncertain situations. In particular, formalisms based on *Bayesian networks* [7] support the representation of almost arbitrary patterns of conditional independence, and algorithms for exploiting whatever independence is expressed in the model.

More specifically, a Bayesian network is a directed acyclic graph representation of a joint probability distribution. For a given set of random variables, the

*This work was supported in part by the Transportation Research Board IVHS-IDEA Program. Contact: wellman@umich.edu

joint distribution represents the probability of each and every combination of possible values. However, Bayesian networks take advantage of conditional independence relations in the distribution to produce a more compact representation. The nodes in the network correspond to the random variables and contain local conditional probability distributions of the corresponding variable given its parents. Each variable is conditionally independent of any non-descendant nodes in the graph given the values of its parent nodes. This information is implicit in the network representation through the absence of links between nodes, and we can exploit the fact to chain together the separate, smaller conditional probability distributions to again recover the joint distribution. Therefore, in principle, we can use the Bayesian network to compute any probability of interest.

In the work described below, we employ Bayesian networks as our basic representation for uncertain states of knowledge about traffic situations. We present specific examples of networks we have developed for recognizing driver maneuvers in traffic monitoring, and for dynamic vehicle control.

3 Plan Recognition

Since driver actions are normally limited to an enumerable set of maneuvers (e.g., lane changes, passing, exiting), it is reasonable to categorize driving at this high level. Recognizing the high-level maneuver being carried out can help in assessing an overall situation and in predicting the future behavior of drivers. To support maneuver recognition (more generally, *plan recognition*), we require a probabilistic model relating the maneuver of a single car to observable features, from which we can categorize maneuvers and predict future behavior given partial information.

To perform general plan recognition tasks, we can generate a Bayesian network representing our causal planning model and use it to support evidential reasoning from observations to plan hypotheses. We begin with a model of the planning agent operating in the world. As it begins planning, the agent has a certain mental state, consisting of its preferences (e.g., goals), beliefs (e.g., about the state of its environment), and capabilities (e.g., available actions). We assume the actual planning process to be some rational procedure for generating the plan that will best satisfy the agent's preferences based on its beliefs, subject to its capabilities. This plan then determines (perhaps with some uncertainty) the actions taken by the agent in the world.

The structure of the Bayesian network is based on the framework depicted in Figure 1. That diagram can itself be viewed as a Bayesian network, albeit with rather broad random variables. To make this operational, we replace each component of the model with a subnetwork that captures intermediate structure for the particular problem. The limited connections among the subnetworks reflect the dependency structure of our generic planning model.

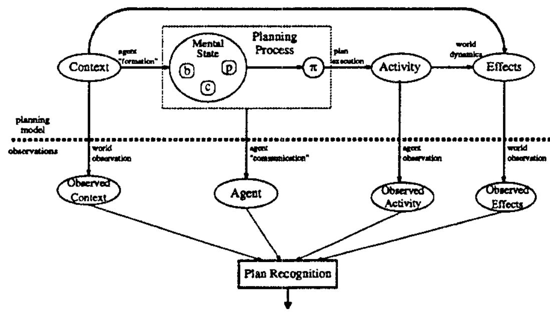

Figure 1: Plan recognition framework.

We applied these plan recognition techniques to the problem of a driver on the highway trying to predict the actions of the other drivers. Since these actions are normally limited to a small set of maneuvers (e.g., lane changes, passing, exiting), recognition of another driver's maneuvering plan would greatly assist in the prediction of future actions. To this end, we have worked on a probabilistic model of the maneuvers of a single car. We can then use this model to identify the current maneuver of an observed car and/or predict future actions, given only partial information.

3.1 Maneuver Recognition Network

Our traffic maneuver recognition network is depicted in Figure 2. A complete description of this network appears elsewhere [9].

This model in effect specifies the likelihood of certain maneuvers under every possible combination of world situation and driver mental state. For example, suppose that the driver is currently traveling below the desired speed and that there is another car directly in front while the lane to the left is clear. Then it is likely that driver will pass the car on the left. This model of the driver's decision process is based in part on the driving model underlying the BAT simulated vehicle, described in Section 4.

3.2 Inference

Once we have created the Bayesian network, we can perform recognition tasks by fixing any observed variables and querying the network about the relevant vari-

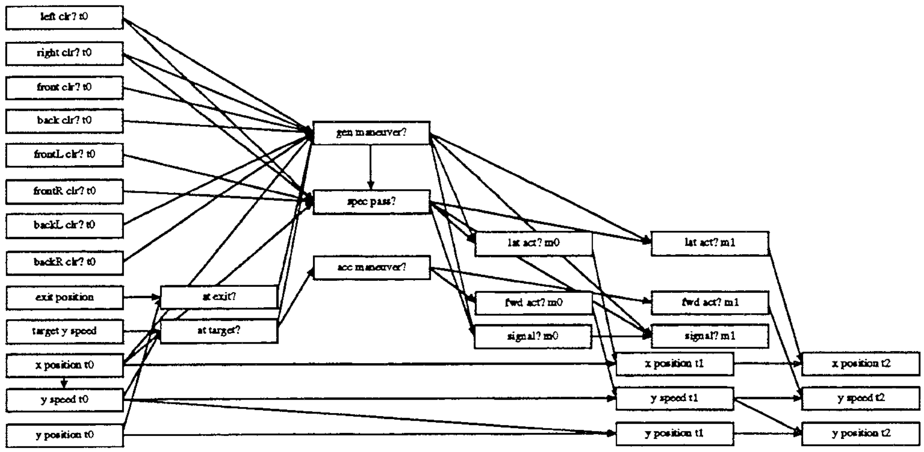

Figure 2: Complete Bayesian network for maneuver recognition.

ables. For instance, we may be interested in determining the plan chosen by the agent, in which case we would examine the nodes in the plan subnetwork. Alternatively, we can predict future agent activity or effects by examining the probabilities of those variables.

Suppose we are trying to predict the behavior of the car behind us as we are driving in the middle lane of a three-lane highway. We observe the car move into the rightmost lane, and we want to determine if it is passing us, preparing to exit, or perhaps simply moving into the slower-moving lane. With respect to the Bayesian network, we have observed that initially front clr? is false and that x position is the middle lane. The only observed effect is that x position at the next point in time is the right lane. If we want to infer the driver's plan, we can examine the general lane change node to see that the posterior probability of a one-lane right shift is 0.64, while that of a pass is 0.35. The former is more plausible since we assume that drivers prefer to pass on the left-hand side, so passing on the right has a relatively low prior probability. The only remaining maneuver with nonzero probability is an exit. All of the other plans have zero probability, since the observed change in lanes violates their definitions.

Given no other contextual observations, it is reasonable to predict that the car will remain in the right lane. However, if there were another car to our left, thus blocking the car behind us from passing on the left, we could also instantiate frontL clr? to be false. We then find that the posterior probability that the car is passing has increased to 0.53, while that for the car simply shifting one lane to the right has dropped to 0.46. The probabilities for the final lane position have changed as well, to 0.51 and 0.48 respectively. Notice that, without knowing about the car to our left, our prediction would be that the car was not passing, but the observation of that aspect of the context changes our belief.

This example illustrates several aspects of our plan-recognition framework, highlighting the importance of accounting for context. By modeling a driver's decision process, observations of the initial state provided strong evidence about the resulting plan. We were also able to model plan execution in a manner similar to other approaches to recognition. The resulting network was able to perform useful inference, even when given only partial observations.

4 Vehicle Control

4.1 The BATmobile

The BAT (*Bayesian Automated Taxi*) project [2] has as its ultimate aim to introduce autonomous vehicles into normal highway traffic. Because the necessary low-level capabilities such as visual vehicle monitoring [3] and lane following [8] are reaching maturity, this challenging problem may not be as intractable as sometimes thought.

The first phase of the project is a feasibility study to establish the computational and sensing requirements for driving and to investigate the nature of the necessary decision algorithms. We use a 2-D physical simulation that generates moderately realistic 3-D rendered video output (SmartPATH), which is passed to the BAT (see Figure 3). Using this information, the BAT must understand the current traffic situation, select high-level actions such as braking, accelerating,

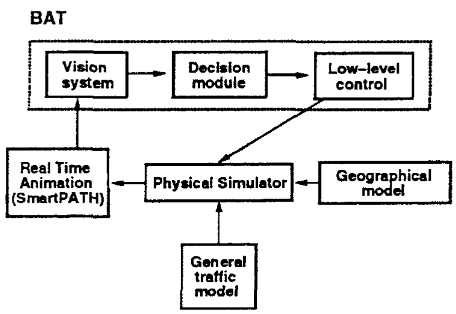

Figure 3: Basic components of the BAT project.

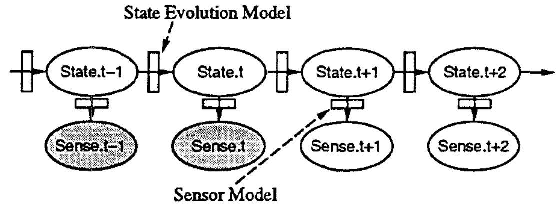

Figure 4: The structure of a dynamic probabilistic network. The ovals denote sets of state nodes or sensor nodes. The arcs going from one slice to the next form the state evolution model, and the arcs going into the sensor nodes form the sensor model. The shaded ovals denote observations available when predicting the state at time $t+1$.

and lane changing, and implement those actions using low-level control.

The BAT's real-time driving task is characterized by pervasive uncertainty:

- noisy sensors—position errors are significant, and some vehicles may not be detected, especially at night or in poor weather conditions,
- sensor inputs must be integrated, and some sensors may fail altogether, and
- the world is only partially observable—vehicles may be occluded, and other drivers' intentions must be inferred (as in Section 3).

4.2 Dynamic Bayesian Networks

To maintain the BAT's belief state, we employ *dynamic Bayesian networks* (DBNs). DBNs are an extension of Bayesian networks that allow variables to take on different values over time [1]. Figure 4 shows the general structure of a DBN. Typically, observations are taken at regular "time slices", and a given network structure is replicated for each slice. DBNs model their domains as partially observable Markov processes, so nodes can be connected not only to other nodes within the same time slice but also (and only) to nodes in the immediately preceding or immediately following slice. The Markov property states that the future is independent of the past given the present. As long as the BAT's representation of the world conforms to this property, the BAT need not maintain the history of its percepts to predict the next state since the accumulated effect of its observations is captured in its current belief state.

As implemented, the BAT monitors each vehicle tracked by the sensor system with a separate DBN. Each network contains nodes for sensor observations, such as vehicle position and velocity, as well as nodes for predicting driver intentions, such as whether the driver intends to make a lane change or to slow down.

Like a Kalman filter, each network computes probability distributions for a vehicle's position and velocity based on both its latest observations and its previous state estimate (which reflects the influence of all previously observed evidence). Unlike a Kalman filter, which is limited to Gaussian distributions, the network predictions can be arbitrarily distributed. For example, if a vehicle were approaching some debris directly in front of it, the network could predict that the vehicle would move either to the right or to the left (but not straight) in order to avoid the debris. Also, the network could easily incorporate additional sensor information. If the sensor system recognized that a vehicle was flashing its right turn signal, the network could make predictions that biased the vehicle's position towards the right.

To incorporate the influence of nearby vehicles, each network contains nodes corresponding to those vehicles. For example, the Front Clear and Front Speed Diff nodes in Figure 5 refer to "the space between this vehicle and the vehicle in front", and "the speed difference between this vehicle and the vehicle in front", respectively. Since the vehicle in front of or behind a given vehicle may change, these *indexical* nodes do not correspond to a specific vehicle. Instead, a preprocessing step using sensor data determines the spatial relationships among the vehicles and then sets the node states accordingly. Figure 5 shows an example vehicle network for one time slice, along with the inter-slice links to the next time slice.

4.3 Decision Making in the BAT

Technically, the decision problem faced by the BAT can be characterized as a partially observable Markov decision process (POMDP). Computing the optimal policy for POMDPs is feasible only for tiny state spaces. Therefore, we have explored three approximation approaches: (1) bounded lookahead using dy-

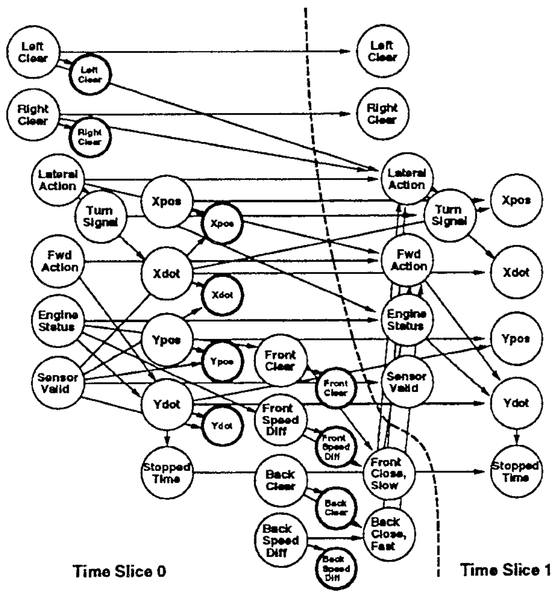

Figure 5: Dynamic Bayesian network for one vehicle, including inter-slice arcs. The smaller nodes with thicker outlines denote sensor observations.

namic decision networks, which incorporate action nodes and an explicit utility function, (2) hand-coded, explicit policy representations, such as decision trees, that take as input the joint probability distribution encoded in the DPN, and (3) supervised learning and reinforcement learning methods for solving the POMDP, in which we learn a policy representation, a utility function on belief states, or an action-value function on belief-state/action pairs. Here we briefly describe the second approach.

The decision tree is a tree of binary if-then-else constructs where the test predicates are computed from the joint distribution computed by the DPN. Each leaf of the tree is a decision. This obviously yields an effective, real-time policy, but constructing the decision tree is a difficult task. Other researchers, for example Lehner and Sadigh [6], have examined the creation of decision trees from Bayesian networks, but only for static problems. In such cases, the decision tree nodes test fully determined evidence variables. If this method is applied to DBNs, one may be forced to test the entire percept sequence.

Our approach involves testing the current belief state instead. Although this is potentially much smaller, optimality in a POMDP requires that the tests define regions in the joint probability space rather than regions in the marginal probability space for each variable. We have found this extremely unintuitive, and so have used tests on marginals of individual variables as an approximation. To date, this has been reasonably effective. We have implemented several hand-constructed decision trees, which have the following general structure (each "predicate" here is actually a complex set of probability thresholds on specific variables, and each "action" a subsidiary decision tree):

```
if changing lane
    if safe to continue
        Continue lane change
    else
        Abort lane change
else if not in target lane and can change lanes
    Initiate lane change to target lane
else if vehicle in front
    Maintain safe following or Pass
else
    Maintain target speed
```

Figure 6: General decision-tree structure.

4.4 Results

We have built a working simulator to test various decision making modules for the BAT. For each test, the simulator reads a scenario description file, which describes the volume of traffic and the behaviors of other vehicles traveling along the highway. At each "clock tick", the simulator determines the trajectories of all the vehicles until the next tick, passing state information in the form of sensor readings (adding noise as necessary using sensor models) to each vehicle's controller, which in turn outputs its decision for the current time step. The simulator uses the vehicle's decision and a physical model to plot trajectories and to detect collisions and other significant events.

The goal of the BAT controller is to maintain a target speed in a target lane. When other vehicles interfere, the controller makes appropriate acceleration/deceleration and lane-changing maneuvers. We show present a scenario where the BAT passes a slow-moving vehicle in Figure 7. We depict the situation as a discrete sequences of 2-D pictures, although of course they are actually continuous 3-D video sequences. In the figure, the BAT is the shaded vehicle.

5 Probabilistic Inference

Using the probabilistic models described above has required several enhancements to existing techniques for inference in Bayesian networks. One important technique is *rollup*, the process of incorporating evidence

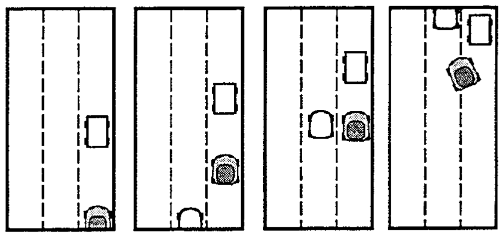

Figure 7: As the BAT approaches a slower vehicle, it decides to pass to the left so that it can maintain its target speed. Because of another vehicle in that lane, the BAT waits until the left lane is clear and then performs a left lane change maneuver and accelerates back to its target speed.

(such as sensor readings) and updating a DBN over time. Any exact method for rolling up one slice of a network is equivalent to performing a sequence of *node eliminations* [5]. Node elimination may introduce additional links into the network, complicating probabilistic inference in several ways. By restricting our structures to *temporally invariant networks* [2], we can alleviate some of these complications.

We have also improved the performance of rollup by enhancing procedures for stochastic simulation in DBNs. Specifically, we have introduced two new techniques, *evidence reversal* and *survival of fittest sampling* [4]. Downstream evidence (i.e., observed nodes with parents) can degrade performance of stochastic simulation. Evidence reversal transforms a given network so that all evidence nodes are at the root. This is a provably correct operation that yields significant improvement in performance, especially in the type of DBNs used in the BAT. Survival of fittest sampling is based on the use of a fixed-size population of samples. After each decision epoch, we extend the sample population by one time slice. We randomize the repopulation process based on the likelihood of the evidence given a particular sample (thus this algorithm is related to genetic algorithms although there is no crossover in our approach).

Stochastic simulation permits us to consider networks with continuous random variables, which often arise in traffic modeling. Another approach that facilitates use of continuous variables is *state-space abstraction* [10], a technique for probabilistic inference at varying levels of granularity. By solving the network using progressively refined state variables, we can achieve whatever level of accuracy time will allow, while still providing useful results even in highly time-stressed situations.

6 Discussion

Although there are dozens of projects worldwide with related goals, descriptions of which can be found in the proceedings of many ITS conferences (including this one), few have taken uncertainty in sensors and actuators as a central characteristic. Among these, the work reported here represents the only effort to date (of which we are aware) to apply Bayesian networks to traffic monitoring or vehicle control. Because Bayesian networks provide a very general, principled, and flexible basis for probabilistic reasoning, and based in part on our initial results, we expect that these methods will receive increasing attention by traffic modelers and other ITS researchers.

References

[1] Thomas Dean and Keiji Kanazawa. Probabilistic temporal reasoning. In *Proceedings of the National Conference on Artificial Intelligence*, pages 524–528, 1988.

[2] Jeffrey Forbes, Timothy Huang, Keiji Kanazawa, and Stuart Russell. The BATmobile: Towards a Bayesian automated taxi. In *Fifteenth International Joint Conference on Artificial Intelligence*, Montreal, 1995.

[3] T. Huang, D. Koller, J. Malik, et al. Automatic symbolic traffic scene analysis using belief networks. In *Proceedings of the National Conference on Artificial Intelligence*, pages 966–972, Seattle, WA, 1994.

[4] Keiji Kanazawa, Daphne Koller, and Stuart Russell. Stochastic simulation algorithms for dynamic probabilistic networks. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, Montreal, 1995.

[5] U. Kjaerulff. A computational scheme for reasoning in dynamic probabilistic networks. In *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence*, pages 121–129, Stanford, CA, 1992.

[6] Paul E. Lehner and Azar Sadigh. Two procedures for compiling influence diagrams. In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, pages 335–341, Washington, DC, 1993.

[7] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.

[8] D. Pomerleau. *Neural Network Perception for Mobile Robot Guidance*. Kluwer, Dordrecht, 1993.

[9] David V. Pynadath and Michael P. Wellman. Accounting for context in plan recognition, with application to traffic monitoring. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, Montreal, 1995.

[10] Michael P. Wellman and Chao-Lin Liu. State-space abstraction for anytime evaluation of probabilistic networks. In *Tenth Conference on Uncertainty in Artificial Intelligence*, Seattle, WA, 1994.