Engng Applic. Artif. Intell. Vol. 4. No. 4, pp. 279–285, 1991
Printed in Great Britain.

0952-1976/91 $3.00 + 0.00

Contributed Paper

Combining Artificial Neural Networks and Symbolic Processing for Autonomous Robot Guidance

DEAN A. POMERLEAU
Carnegie Mellon University, Pittsburgh

JAY GOWDY
Carnegie Mellon University, Pittsburgh

CHARLES E. THORPE
Carnegie Mellon University, Pittsburgh

Artificial neural networks are capable of performing the reactive aspects of autonomous driving, such as staying on the road and avoiding obstacles. This paper describes an efficient technique for training individual networks to perform these reactive driving tasks. But driving requires more than a collection of isolated capabilities. To achieve true autonomy, a system must determine which capabilities should be employed in the current situation to achieve its objectives. Such goal-directed behavior is difficult to implement in an entirely connectionist system. This paper describes a rule-based technique for combining multiple artificial neural networks with map-based symbolic reasoning to achieve high-level behaviors. The resulting system is not only able to stay on the road, it is able to follow a route to a predetermined destination, turning appropriately at intersections and stopping when it has reached its goal.

Keywords: Autonomous guided vehicles, intelligent guidance, neural net guidance, symbolic processed maps, vehicle motion control, modular architectures, ALVINN.

INTRODUCTION

Artificial neural networks are commonly employed as monolithic non-linear classifiers. The technique, often used in domains such as speech, character and target recognition, is to train a single network to classify input patterns by showing it many examples from numerous classes. The mapping function from inputs to outputs in these classification tasks can be extremely complex, resulting in slow learning and unintelligible internal representations.

However, there is an alternative to this monolithic network approach. By training multiple networks on different aspects of the task, each can learn relatively quickly to become an expert in its sub-domain. This paper describes a technique we have developed to quickly train expert networks for vision-based autonomous vehicle control. Using this technique, specialized networks can be trained in under 5 min to drive in situations such as single-lane road driving, highway driving, and collision avoidance.

Achieving full autonomy requires not only the ability to train individual expert networks, but also the ability to integrate their responses. This paper focuses on rule-based arbitration techniques for combining multiple driving experts into a system that is capable of guiding a vehicle in a variety of circumstances. These techniques are compared with other neural network integration schemes and shown to have a distinct advantage in domains where symbolic knowledge and techniques can be employed in the arbitration process.

DRIVING MODULE ARCHITECTURE

The architecture for an individual ALVINN driving module is shown in Fig. 1. The input layer consists of a single 30×32 unit "retina" onto which a sensor image from either the video camera or the laser range finder is projected. Each of the 960 input units is fully connected to the hidden layer of 5 units, which is in turn fully connected to the output layer. The 30-unit output layer is a linear representation of the currently appropriate

Correspondence should be sent to: Dean A. Pomerleau: School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213. U.S.A.

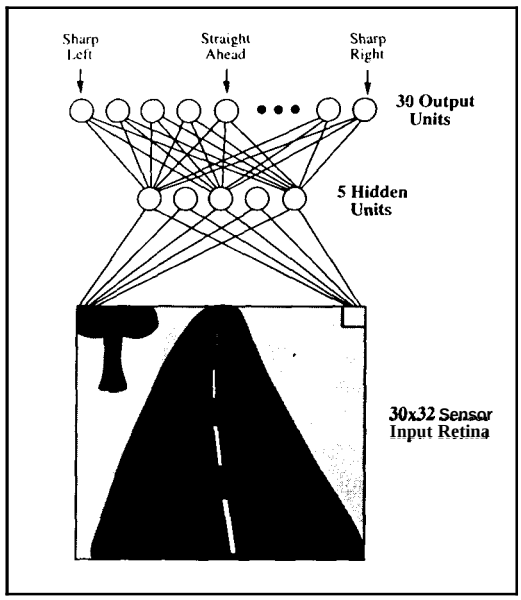

Fig. 1. The architecture for an individual ALVINN driving module.

steering direction. The centermost output unit represents the "travel straight ahead" condition, while units to the left and right of center represent successively sharper left and right turns. The steering direction dictated by the network may serve to keep the vehicle on the road or to prevent it from colliding with nearby obstacles, depending on the type of sensor input and the driving situation the network has been trained to handle.

To drive the Navlab, an image from the appropriate sensor is reduced to 30 x 32 pixels and projected onto the input layer. After propagating activation through the network, the output layer's activation profile is translated into a vehicle steering command. The steering direction dictated by the network is taken to be the center of mass of the "hill" of activation surrounding the output unit with the highest activation level. Using the center of mass of activation instead of the most active output unit when determining the direction to steer permits finer steering corrections, thus improving ALVINN's driving accuracy.

INDIVIDUAL DRIVING MODULE TRAINING AND PERFORMANCE

We have developed a scheme called training "on-the-fly" to quickly teach individual modules to imitate the driving reactions of a person. As a person drives, the network is trained with back-propagation using the latest video camera image as input and the person's current steering direction as the desired output. To facilitate generalization to new situations, additional variety is added to the training exemplars by shifting and rotating the original camera image in software to make it appear that the vehicle is situated differently relative to the environment (See Fig. 2). The correct

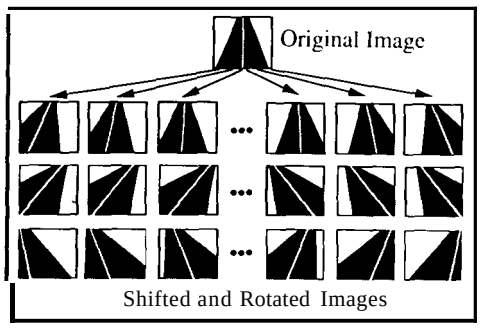

Fig. 2. The single original video image is shifted and rotated to create multiple training exemplars in which the vehicle appears to be at different locations relative to the road.

steering direction as dictated by the driver for the original image is altered for each of the transformed images to account for the altered vehicle placement. Adding these transformed patterns to the training set allows the network to learn to recover from driving mistakes, without requiring the human trainer to explicitly stray from the road center and then return. For more details about the technique and purpose of training on-the-fly see Ref. 1.

Running on three Sun-4 Sparcstations, training on-the-fly requires about 5 min during which a person drives the Navlab at about 6 mph over a 1/4 to 1/2 mile stretch of training road. Once it has learned, the network can accurately traverse the length of road used for training and also generalize to drive along parts of the road it has never encountered, under a variety of weather conditions. In addition, since determining the steering direction from the input image merely involves a forward sweep through the network, the system is able to process 20 images per second, allowing it to drive at up to the Navlab's maximum speed of 20 mph.* This is over twice as fast as any other sensor-based autonomous system has driven the Navlab.[2,3]

The flexibility provided by training on-the-fly has facilitated the development of individual driving networks to handle numerous situations. Using video camera images as input, networks have been trained to drive on single-lane dirt roads, single-lane paved roads, two-lane suburban neighborhood streets, and lined two-lane highways (See Fig. 3).

By replacing the video input with alternative sensor modalities. ALVINN has learned other interesting behaviors. One such sensor onboard the Navlab is a scanning laser range finder. The range finder provides images in which pixel values represent the distance from the range finder to the corresponding area in the scene. Obstacles such as trees and cars appear as discontinuities in depth, as can be seen in the range finder image in Fig.4. Using this sensor, separate ALVINN modules have been trained to avoid collisions

*The Navlab has a hydraulic drive system that **allows** for very precise speed control, but that prevents the vehicle from driving over 20 mph.

Fig. 3. Video images taken on three of the roads ALVINN modules have been trained to handle. They are, from left to right, a single-lane dirt access road, a single-lane paved bicycle path, and a lined two-lane highway.

in obstacle-rich environments and to follow alongside rows of parked cars.

A third type of image used as input to ALVINN modules comes from a laser reflectance sensor. In this type of image, a pixel's value corresponds to the amount of laser light that reflects off the corresponding point in the scene and back to the sensor. The road and off-road regions reflect differently, making them distinguishable in the image (see Fig. **4).** Laser reflectance images in many ways resemble black and white video images, but have the advantage of being independent of ambient lighting conditions. Using this sensor modality, we have trained a network to follow single-lane roads in total darkness.

SYMBOLIC KNOWLEDGE AND REASONING

Despite the variety of capabilities exhibited by individual driving networks, until recently the system has been far from truly autonomous. First, the one driving network architecture shown in Fig. 1 was capable of driving only on the type of road on which it was trained. If the road characteristics changed, ALVINN would often become confused and stray from the road. In addition, a real autonomous system needs to be capable of planning and traversing a route to a goal. The neural network driving modules are good at reactive tasks such as road following and obstacle avoidance, but the networks have a limited capability for the symbolic tasks necessary for an autonomous mission. The system of networks cannot decide to turn left at an intersection in order to reach a goal. After making a turn from a one-lane road to a two-lane road, the system does not know that it should stop listening to one network and start listening to another. Just as a human needs symbolic reasoning to guide reactive processes, the networks need a source of symbolic knowledge to plan and execute a mission.

Ideally, the symbolic knowledge source would reason like a person. It would use its knowledge of the world to plan a sequence of observations and corresponding actions to traverse the route. For instance, to achieve the goal of reaching a friend's house, the mission description might be a sequences like, "Drive until the sign for Seneca Road is seen, and turn left at that intersection. Then drive until the third house on the left is seen, and stop in front of it."

In this ideal system, once the mission is planned, the symbolic knowledge source would rely entirely on perception to control the execution of the mission. In other words, the symbolic resource module would be able to recognize events and use what it sees to guide the interaction of the networks. The symbolic resource module would be capable of reading the street sign at an intersection and making the appropriate turn to continue on to its destination. It would also be able to identify the new road type and choose the appropriate network for driving on that kind of road. Unfortunately, the perception capabilities required by such a module are beyond the current state of the art.

In order to bridge the gap between mission requirements and perception capabilities, we use additional geometric and symbolic information stored in an "annotated map". An annotated map is a 2-D data structure containing geometrical information about the area to be traversed, such as the locations of roads and landmarks. In addition, each object in the map can be annotated with extra information to be interpreted by the clients that access the map. For example, as far as the annotated map is concerned, a mailbox is simply a

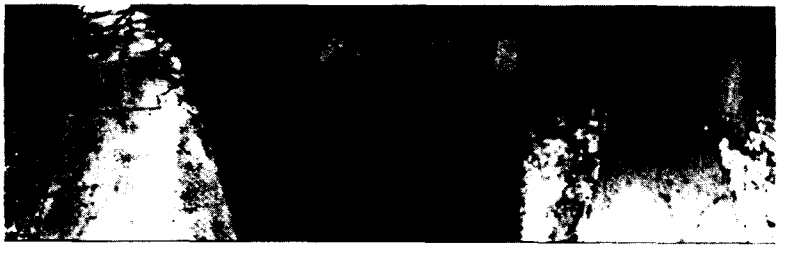

Fig. **4.** Images taken of the scene using the three sensor modalities the system employs as input. From left to right they are a video image, a laser range finder image, and a laser reflectance image. Obstacles like trees appear as discontinuities in laser range images. The road and the grass reflect different amounts of laser light, making them distinguishable in laser reflectance images.

2-D polygon at a particular location with some extra bits associated with it. The "extra bits" might represent the **3-D** shape of the mailbox, or even the name of the person who owns it. The module which manages the annotated map does not interpret this extra information, but rather provides a mechanism for client modules to access the annotations. This reduces the knowledge bottleneck that can develop in large, completely centralized systems.

The annotated map is not just a passive geometric database, but instead is an active part of our system. Besides having a 2-D representation of the physical objects in a region, annotated maps can contain what are called "alarms". Alarms are conceptual objects in the map, and can be lines, circles, or regions. Each alarm is annotated with a list of client modules to notify and the information to send to each when the alarm is triggered. When the annotated map manager notices that the vehicle is crossing an alarm on the map, it sends the information to the pertinent modules. Once again, the map manager does not interpret the information: that is up to the client modules.

Alarms can be thought of as positionally based production rules. Instead of using perception based production rules like, "If A is observed, then perform action B", an annotated-map-based system has rules of the form, "If location A is reached, then perform action B". Thus we reduce the problem of making high-level decisions from the difficult task of perceiving and reacting to external events to the relatively simple task of monitoring and updating the vehicle's position.

The first step in building an annotated map is collecting geometric information about the environment. We build our maps by driving the vehicle over roads and linking the road segments together at intersections. At the same time, a laser range finder is used to record the positions of landmarks such as mailboxes and telephone poles. Planning a particular mission requires adding specific instructions to the map in the form of "trigger annotations". This is currently a process performed by the person planning the mission. For example. the human expert knows that when approaching an intersection, the vehicle should slow down, so the expert chooses the appropriate location to put the trigger line. The trigger line goes across the road at the point, and is annotated with a string of bits that represents the new speed of the vehicle. During the run, when the vehicle crosses the trigger line, the map manager sends the string of bits to a module that interprets the information and slows the vehicle to the desired speed. In the current system, alarms are interpreted as commands, but there is no predefined "correct" way for a module to react to an alarm. Depending on its content, an alarm could also be interperted as a wakeup **call,** or even as simply advice.

Because position information is so critical to an annotated map system, we use multiple techniques to determine the vehicle's current location. We use an

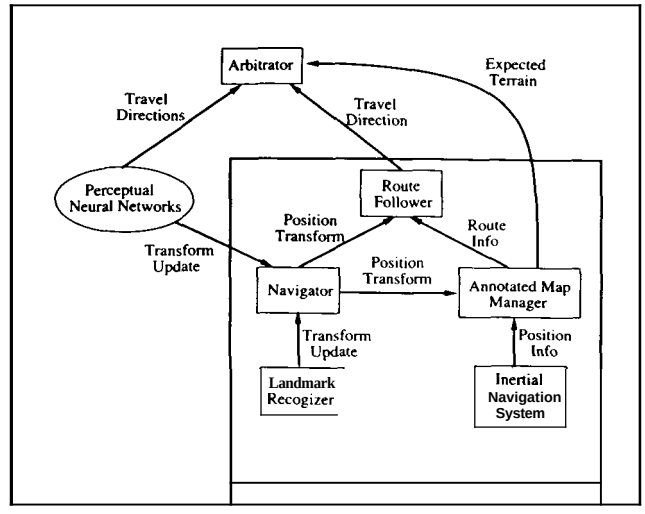

Fig. 5. The components of the annotated map system and the interaction between them. The annotated map system keeps track of the vehicle's position on a map. It provides the arbitrator with symbolic information concerning the direction to steer to follow the preplanned route and the terrain the vehicle is currently encountering. The neural network driving modules are condensed for simplicity into a single block labeled "perceptual neural networks".

Inertial Navigation System (INS) which can determine the vehicle's location with an error of approximately 1% of distance traveled.[4] To eliminate the positioning error that accumulates over time in the INS data, the annotated map system also uses information from perception modules. For example, since the driving networks presumably keep the vehicle on the road, lateral error in the vehicle positioning system relative to the road can be identified and eliminated. In addition, a module using the laser range finder compares the landmarks it sees to the landmarks collected when the map was built, and triangulates the vehicle's position on the map. These techniques allow perception modules to provide useful positioning information ***without*** requiring them to explicitly recognize and interpret particular objects such as street signs. The position corrections provided by perception modules are interpreted as a change in the transform between the location that the INS reports and the real vehicle position on the map. A separate module, called the navigator, is in charge of maintaining and distributing this position transform.

Annotated maps provide the system with the symbolic information and control knowledge necessary for a fully autonomous mission. Since the control knowledge is geometrically based, and since planning is done before the mission starts, runtime control comes at a low computational cost. Figure 5 shows the structure and interaction of the annotated map system's components. It also illustrates the annotated map system's interaction with the other parts of the system, including the perceptual neural networks and the arbitrator (discussed below). Figure 6 shows a map and annotations for a mission segment.

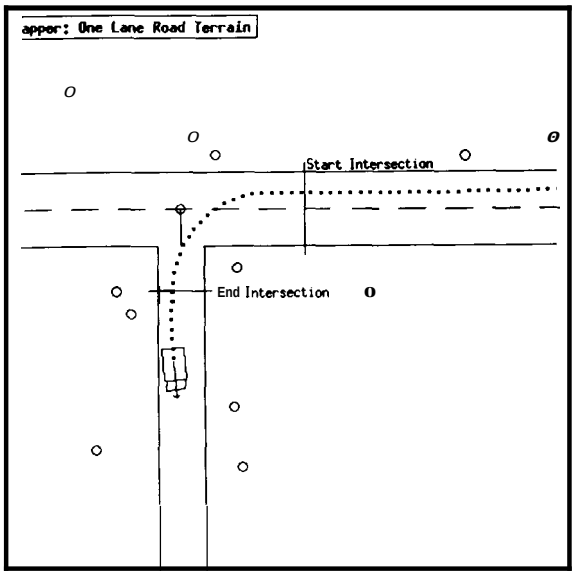

Fig. 6. A section of the map created and maintained by the annotated map system. The map shows the vehicle traversing an intersection between a single- and a two-lane road. The lines across the roads are alarms which are triggered when crossed by the vehicle. Triggering an alarm results in a message being passed from the map manager to the arbitrator indicating a change in terrain type. The circles on the map represent the positions of landmarks, such as trees and mailboxes. The annotated map system uses the locations of known landmarks to correct for vehicle positioning errors which accumulate over time.

RULE-BASED DRIVING MODULE INTEGRATION

We use the symbolic knowledge provided by the annotated map system to help guide the interaction of the reactive driving neural networks. Figure 7 shows the system architecture with emphasis on the neural networks. Whereas Fig. 5 subsumed the neural network systems into one unit labeled "perceptual neural networks", Fig. 7 subsumes the annotated map system into one package. In this diagram, each box represents a separate process running in parallel.

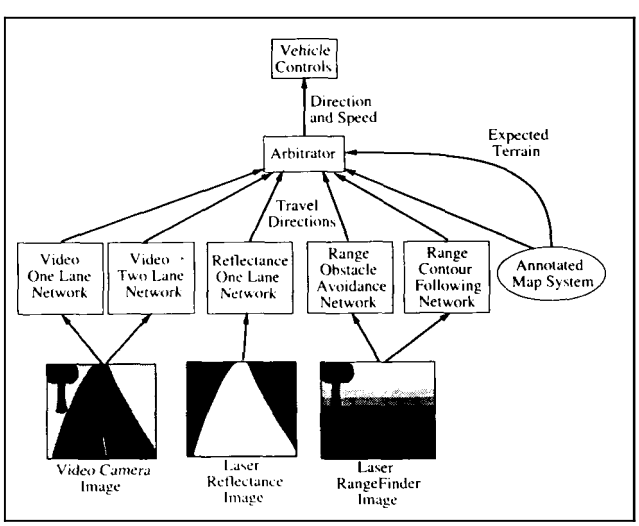

Fig. 7. The integrated ALVINN architecture. The arbitrator uses the terrain information provided by the annotated map system, as well as symbolic models of the driving networks' capabilities and priorities, to determine the appropriate module lor controlling the vehicle in the current situation.

Images from the three onboard sensors are provided to the five driving networks shown in the second row of the diagram. The driving networks propagate activation forward through their weights, with each determining what it considers to be the correct steering direction. These steering directions are sent to the arbitrator, which has the job of deciding which network to attend to and therefore how to steer.

The arbitrator makes use of both the geometric and control information provided by the annotated map system to perform a mission autonomously. First, the route-following module within the annotated map system uses the geometric information in the annotated map to recommend a vehicle steering direction. The direction recommended by the route follower is the direction it thinks the vehicle should steer in order to follow the preplanned route. When the vehicle is driving down a road, the route follower queries the annotated map for the position of the road ahead of the vehicle. The route follower uses this geometric information to generate a steering direction.

The annotated map system also provides the arbitrator with information about the current driving situation, including what type of road the vehicle is on, and whether there is an intersection or dangerous permanent obstacle ahead. For example, suppose during the planning phase the human expert notices that at a particular point the road changes from one lane to two. The expert would set a trigger line at the corresponding point on the map and annotate it with a message that will tell the arbitrator to stop listening to the one-lane road-following network and start listening to the two-lane road-following network. When the alarm is triggered during the run, the arbitrator combines the advice from the annotated map system with the steering directions of the neural network modules using a technique called "relevancy arbitration".

Relevancy arbitration is a straightforward idea. If the annotated map system indicates the vehicle is on a two-lane road, the arbitrator will steer in the direction dictated by the two-lane road driving network, since it is the relevant module for the current situation. If the annotated map system indicates the vehicle is approaching an intersection, the arbitrator will choose to steer in the direction dictated by the annotated map system, since it is the module that knows which way to go in order to head towards the destination. In short, the arbitrator combines symbolic knowledge of driving module capabilities with knowledge of the present terrain to determine the relevant module for the current circumstances.

The relevancy of a module need not be based solely on the current terrain information provided by the annotated map system. Instead, the arbitrator also employs rules for determining a module's relevancy from the content of the module's message. The obstacle avoidance network has one such rule associated with it. The obstacle avoidance network is trained to steer

straight when the terrain ahead is clear and to swerve to prevent collisions when confronted with obstacles. The arbitrator gives low relevancy to the obstacle avoidance network when it suggests a straight steering direction, since the arbitrator realizes it is not an applicable knowledge source in this situation. But when it suggests a sharp turn, indicating there is an obstacle in the vehicle's path, the urgency of avoiding a collision takes precedence over other possible actions, and the steering direction is determined by the obstacle avoidance network. This priority arbitration is similar in many ways to the subsumption architecture,' although the most common interaction between behaviors in Brooks' systems is for higher-level behaviors to override less sophisticated, instinctual ones.

By combining map-related knowledge about the current driving situation with knowledge about abilities and priorities of individual driving modules, the integrated architecture provides the system with capabilities that far exceed those of individual driving modules alone. Using this architecture, the system has successfully followed a 1/2-mile path through a suburban neighborhood from one specific house to another. In navigating the route, the system was required to drive through 3 intersections onto 3 different roads while swerving to avoid parked cars along the way. At the end, the vehicle came to rest 1m from its destination.

ANALYSIS AND DISCUSSION

Rule-based integration of multiple expert networks has significant advantages over previously developed connectionist arbitration schemes. One such advantage is the ease of adding new modules to the system. Using rule-based arbitration, the new module can be trained in isolation to become an expert in a new domain, and then integrated by writing rules for the arbitrator which specify the new module's area of expertise and its priority. This is in contrast to other connectionist expert integration techniques, such as the task decomposition architecture: connectionist glue' and the meta-pi architecture.[8,9] To combine experts using these techniques requires the training of additional neural network structures, either simultaneously with the training of the experts, in the case of the task decomposition architecture, or after expert training in the case of the connectionist glue and meta-pi architectures. Adding a new expert using the techniques requires retraining the entire integrating structure from scratch, which involves presenting the system patterns from each of the experts' domains, not just the new one. This large-scale retraining is particularly difficult in a task like autonomous navigation because it requires either driving over all the experts' domains again, or storing a large number of domain-specific images for later reuse.

Another significant advantage of rule-based arbitration is the ease with which non-neural-network knowledge sources can be integrated into the system. Symbolic tasks such as planning and reasoning about a map are currently difficult to implement using neural networks. In the future, it should be possible to implement more and more symbolic processing using connectionist techniques, but until then, rule-based arbitration provides a way of bridging the gap between neural networks and traditional AI systems.

The technique is not without shortcomings, however. The current implementation relies too heavily on the accuracy of the annotated map system, particularly for negotiating intersections. The question might be asked, why is the mapping system required for intersection traversal in the first place? Why can't the driving networks handle intersections? The answer is that when approaching an intersection, an individual driving network will often provide ambiguous steering commands, since there are multiple possible roads to follow. If left on its own, a road-following network will often alternately steer towards one or the other road choices, causing the vehicle to oscillate and eventually drive off the road. In addition, even if the network could learn to definitively choose one of the branches to follow, it still would not know which is the ***appropriate*** branch to choose in order to head toward the destination. In short, the mapping modules can be viewed both as a useful source of high-level symbolic knowledge, and as an interim solution to the difficult perceptual task of intersection navigation.

The annotated map system as currently implemented is not a perfect solution to the problem of high-level guidance because it requires both detailed knowledge of the route, and an accurate idea of the current vehicle position. In certain controlled circumstances, such as rural mail delivery, the same route is followed repeatedly, making an accurate map of the domain feasible. However, a system capable of following less-precise directions, like "go about a half mile and turn left on Seneca Road", is clearly desirable. Such a system would require more reliance on observations from perception modules and less reliance on knowledge of the vehicle's exact position when making high-level decisions.

Conceptually, this shift towards reliance on perception for high-level guidance could be done in two ways. First, observations of objects like the Seneca Road street sign, could be used to update the vehicle's position on the map. In fact, position updates based on perceptual observations are currently employed by the annotated map system when it triangulates the vehicle's location based on the positions of known landmarks in laser range images. But position updates are only helpful when the observations are location-specific. For observations of objects like stop lights, or arbitrarily located objects like "road construction ahead" signs, the system's response should be independent of the vehicle's location.

These location-independent observations could be

modeled as positionless alarms in the annotated map. When a perception module sees an object like a "road construction ahead" sign, it would notify the map manager. The map manager would treat the sighting as an alarm, distributing the information associated with the alarm to the pertinent modules. Perception triggered alarms would allow the system to transition between its current perceptual abilities and future, more-advanced capabilities.

Although the system is not yet capable of identifying and reading individual signs, we have had preliminary success in using neural network perceptual observations to help guide high-level reasoning. The technique relies on the fact that when the vehicle reaches an intersection, the output of the driving network becomes ambiguous. This ambiguity manifests itself as an output vector with more than one active steering direction corresponding to the multiple possible branches to follow. This output ambiguity has been successfully employed to update the vehicle's position and to follow coarse directions. As the vehicle approaches an intersection, the annotated map system signals the arbitrator that an intersection is coming up and that the vehicle should follow the right-hand branch in order to head towards the goal. This level of detail does not require either a highly accurate map or precise knowledge of the vehicle's current position. The arbitrator takes the annotated map system's message as a signal to watch the output of the current driving network carefully. When the driving network's output becomes ambiguous, the arbitrator signals the annotated map system that the vehicle has reached the intersection and to update the vehicle's position accordingly. The arbitrator also uses the "turn right" portion of the annotated map system's message in order to choose the correct steering direction from the driving network's ambiguous output vector. This closer interaction between the perception networks and the annotated map allows the system to use perception for intersection traversal, instead of relying solely on knowledge from the map for guidance.

Another shortcoming of rule-based arbitration as currently implemented is its binary nature. Currently, a module is deemed by the annotated map system as either appropriate or inappropriate for the current road type. This binary decision does not address the question of intelligently combining modules trained for the same domain, such as the video-based single-lane driving network and the laser reflectance-based single-lane driving network. There are obviously some situations, such as night driving, when one network is better suited than the other. To take more subtle circumstances into account when weighting the steering directions dictated by multiple networks, we are developing augmented arbitration rules that consider more context than just the current road type. We are also currently working on connectionist techniques that can determine a network's reliability directly from its output alone. Preliminary results in this area look very promising.

One final drawback of the current system is the need for a human expert to preplan the mission by providing map annotations. In the future, we will replace the human expert with an expert system capable of annotating the map appropriately. We understand the techniques the human expert uses to find the shortest route and to annotate the map, so automating the process should not be difficult.

In conclusion, a modular architecture permits rapid development of expert neural networks for complex domains like autonomous navigation. Rule-based arbitration is a simple and efficient method for combining these experts when symbolic knowledge is available for reasoning about their appropriateness. Rule-based arbitration also permits the combination of neural network experts with non-neural network processing techniques such as planning, which are difficult to integrate using other arbitration schemes.

Acknowledgements—This work would not have been possible without the input and support provided by Dave Touretzky, John Hampshire, and especially Omead Amidi, James Frazier and rest of the CMU ALV group.

The principle support for the Navlab has come from DARPA, under contracts DACA76-85-C-0019, DACA76-85-C-0003 and DACA76-85-C-0002. This research was also funded in part by a grant from Fujitsu Corporation.

REFERENCES

1. Pomerleau D. Efficient training of artificial neural networks for autonomous navigation. In ***Neural Computation,*** Vol. ***3:*** 1. (Sejnowski T., Ed.). MIT Press, Cambridge, MA (1991).
2. Kluge K. and Thorpe C. Explicit models for robot road following. In ***Vision and Navigation: The CMU Navlab*** (Thorpe C., Ed.). Kluwer Academic, Boston (1990).
3. Crisman J. D. and Thorpe C. Color vision for road following. In ***Vision and Navigation: The CMU Navlab*** (Thorpe C., Ed.). Kluwer Academic, Boston (1990).
4. Amidi O. and Thorpe C. Integrated mobile robot control. ***Proc. SPIE Mobile Robots*** V 1388,504–524 (1990).
5. Brooks R. A. A robust layered control system for a mobile robot. ***IEEE 1. Robotics Autom.*** RA-2, (1), 14–23 (1986).
6. Jacobs R. A., Jordan M. I. and Barto A. G. Task decomposition through competition in a modular connectionist architecture: the what and where vision tasks. University of Massachusetts, Computer and Information Science Technical Report 90-27, March (1990).
7. Waibel A. Modular construction of time delay neural networks for speech recognition. In ***Neural Computation,*** Vol. 1:1. MIT Press, Cambridge, MA (1989).
8. Hampshire J. B. and Waibel A. The meta-pi network: building distributed knowledge representations for robust pattern recognition. Carnegie Mellon Technical Report CMU-CS-89-166-R, August (1989).
9. Thorpe C and Gowdy J. Annotated maps for autonomous land vehicles. In ***Proc.*** DA***RPA Image Understanding Workshop,*** pp. 765–771 (1990).