Evolving a Real-World Vehicle Warning System

Nate Kohl
Department of Computer Sciences
University of Texas at Austin
1 University Station, C0500
Austin, TX 78712-0233
nate@cs.utexas.edu

Kenneth Stanley
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, Florida 32816
kstanley@cs.ucf.edu

Risto Miikkulainen
Department of Computer Sciences
University of Texas at Austin
1 University Station, C0500
Austin, TX 78712-0233
risto@cs.utexas.edu

Michael Samples
Center for the Study of Complex Systems
University of Michigan
Ann Arbor, MI 48109
msamples@umich.edu

Rini Sherony
Technical Research Department
Toyota Technical Center
Ann Arbor, MI 48105
rini.sherony@tema.toyota.com

ABSTRACT

Many serious automobile accidents could be avoided if drivers were warned of impending crashes before they occur. Creating such warning systems by hand, however, is a difficult and time-consuming task. This paper describes three advances toward evolving neural networks with NEAT (NeuroEvolution of Augmenting Topologies) to warn about such crashes in real-world environments. First, NEAT was evaluated in a complex, dynamic simulation with other cars, where it outperformed three hand-coded strawman warning policies and generated warning levels comparable with those of an open-road warning system. Second, warning networks were trained using raw pixel data from a simulated camera. Surprisingly, NEAT was able to generate warning networks that performed similarly to those trained with higher-level input and still outperformed the baseline hand-coded warning policies. Third, the NEAT approach was evaluated in the real world using a robotic vehicle testbed. Despite noisy and ambiguous sensor data, NEAT successfully evolved warning networks using both laser rangefinders and visual sensors. The results in this paper set the stage for developing warning networks for real-world traffic, which may someday save lives in real vehicles.

Categories and Subject Descriptors: I.2.6 [LEARNING]: Connectionism and neural nets

General Terms: Experimentation

Keywords: neuroevolution, vehicle, real world, NEAT

Track: Real-World Applications

1. INTRODUCTION

For the average person, the drive to work may be the most dangerous thing they do all day: in 2004, driving-related accidents in the U.S. were responsible for more than 104 deaths and 5,101 injuries *every day*, with an estimated yearly economic cost of $230.6 billion [1]. While the proliferation of automobiles over the last century has played a pivotal role in advancing society, the improvements have clearly not come without a cost.

If cars could warn their drivers that a crash is imminent, it is possible that many accidents could be avoided. One approach for building such a warning system is to ask an expert to describe as many dangerous situations as possible and formalize that information in an automated reasoner. However, such expert systems may fail when exposed to subtle changes in noisy sensor data. Moreover, it may not be possible to predict a crash from a static snapshot of the road; the recent history of the car and other objects on the road should be taken into account as well. It is difficult to know how long such a history should be or what objects it should track.

Yet if the car could learn *on its own* what objects to track and how long to keep salient events in memory, these challenges could be overcome. In addition, cars could be trained with different drivers under different circumstances, creating more flexible warning systems.

Teaching a car to predict crashes is the goal of the automobile warning system project at the University of Texas at Austin, started in November 2003 and funded in part by Toyota. The NeuroEvolution of Augmenting Topologies (NEAT; [9, 10]) method was used to evolve crash prediction neural networks. NEAT is a natural choice for the learning method because it evolves the network topology in addition to the weights, and therefore can develop arbitrary recurrent neural networks that keep a variable length of prior history in memory. In other words, an expert does not need to decide how long the warning window should be or what it should take into account, because evolution makes this determination in selecting the appropriate recurrent topology. In addition, NEAT has shown promise in control tasks that involve noisy, continuous inputs such as pole balancing and a simulated robot duel, suggesting it could make effective judgments about the danger of vehicle circumstances [9, 10]. Because NEAT matches the complexity of the neural network with the complexity of the task, it can find the right level of representation for warning under different conditions. Teaching networks to predict crashes does offer new challenges for NEAT, however, given the potential for a large number of real-world inputs.

In an earlier version of this work, NEAT was shown to be effective at evolving warning systems in simulated open road driving where only collisions with the edge of the road occurred [7]. This paper extends these results in three ways that suggest that this approach will be effective in the real world. The first extension is to include other moving cars in the simulation. The warning task is much more difficult and realistic because it requires predicting the behavior of several moving objects at once. The warning networks that NEAT generates are evaluated empirically and found to perform better than several hand-coded strawman warning policies.

The second extension demonstrates that the warning networks can work with simple visual input. Instead of using expensive and delicate laser rangefinders, simulated low-resolution digital camera data is fed into the evolving networks. Digital cameras are cheap, easy to work with, and provide a veritable deluge of information, making them practical for real-world applications. However, visual data can be very ambiguous and is often of much higher dimensionality than readings from a laser rangefinder. Surprisingly, NEAT performs almost equally well with visual data, and again outperforms the hand-coded baseline policies.

The third contribution is a real-world evaluation of the approach using a robotic vehicle testbed. The lessons learned from the above experiments are applied to the task of training a warning system for a real robot that interacts with a cluttered training environment. Preliminary results suggest that NEAT is able to generate successful warning networks despite noisy real-world training data. These three results set the stage for developing warning networks for real-world traffic, which may someday save lives in real vehicles.

The next section describes the RARS driving simulator used in the simulation experiments and the NEAT neuroevolution method used to train warning networks. Section 3 describes how NEAT performs with other cars. Section 4 compares warning networks evolved with laser rangefinder data to networks evolved with raw pixel data from a camera. Section 5 describes a real-world implementation of this approach in a robotic vehicle testbed, and evaluates future prospects at transferring the system to real cars.

2. BACKGROUND

This section describes the simulator (the Robot Auto Racing Simulator, or RARS), the learning algorithm (NeuroEvolution of Augmenting Topologies, or NEAT), and the experimental method by which NEAT was used to train warning networks.

2.1 The Robot Auto Racing Simulator (RARS)

Since learning requires experience, it is necessary for the learning system to gain experience through driving and predicting crashes. Because crashing cars in the real world with an untested approach would be dangerous and expensive, it is necessary to start in simulation. RARS[1] (Figure 1), a public domain racing simulator designed for testing Artificial Intelligence (AI) methods for real-time control, is ideally suited for this purpose.

RARS is supported by an active community that provides documentation and software maintenance. The software was written with AI in mind, so it is easy to modify existing drivers and introduce new ones. Vehicle dynamics are accurately simulated, including skidding and traction. Multiple automobiles controlled by different automated drivers can interact in the environment at the same time. The software automatically provides information like the distance between the driver and other vehicles and the direction of the road that can be used as the basis for simulated sensors.

The basic type of sensor used for the experiments in this paper is a simulated laser rangefinder, which provides both information about the edge of the road and the location of other cars. Two sets of laser rangefinder data are generated. One set casts seven beams forwards from the front of the car that stop at the edge of the road, giving the car an indication of its position and heading relative to the sides of the road, and also of the curvature of the road. The second set consists of six wide beams that stop when they hit an obstacle on the road, like a car (Figure 2).

It is sensible to ask whether this sensor configuration is a reasonable approximation of the real world, i.e. can similar information be extracted from real sensors? In fact, significant research has gone into vehicle trackers and detecting lanes in the real world [2, 5, 11]. Data from such systems could be processed and fed into the neural networks in a similar form to the sensors used in these experiments. However, such sensors are difficult to build and maintain, which is why visual sensors will also be tested in this paper (Section 4).

RARS provides a virtual gas pedal, brake, and steering wheel that can receive their values from the outputs of a neural network. The gas pedal and brake are interpreted as a requested tire speed relative to the bottom of the car. There is no limit to how high the request can be, and RARS tries to match the request within the physical constraints of the car. If the request is lower than the current speed, RARS attempts to slow the car down by braking. The steering request is treated similarly: the lateral force generated by the same turn angle request increases the higher the current speed. Thus, the driving controls in RARS work like a real automobile, and excessive inputs cause the car to lose traction and skid or spin out of control.

2.2 Neuroevolution of Augmenting Topologies (NEAT)

It is not known how complex a warning neural network needs to be or even what kind of topology it should have. Searching in too large a space, i.e. a space of highly complex networks, would be intractable while searching in too simple a space would limit solution quality. Moreover, it is not known which recurrent connections should exist in the network to allow it to react to past states. For example, if the car is skidding to one side, each snapshot of the sensory input looks perfectly normal. A skid can only be identified by combining the observations over the last several time steps.

The NeuroEvolution of Augmenting Topologies (NEAT) method [9], which automatically evolves network topology

[1]http://rars.sourceforge.net

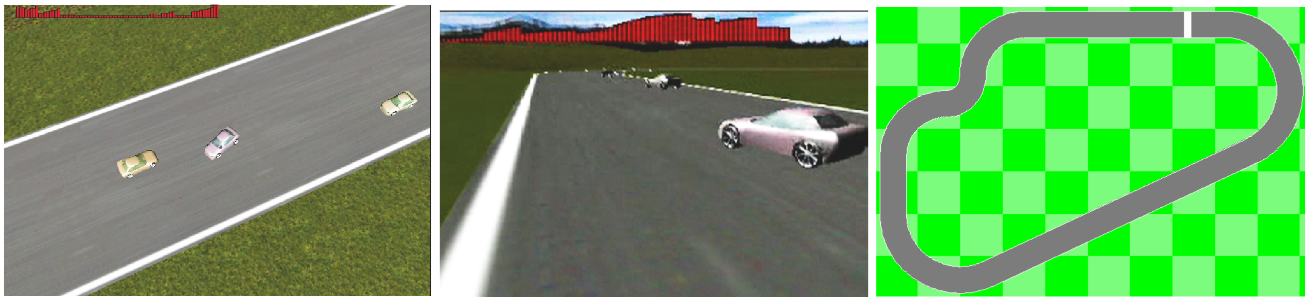
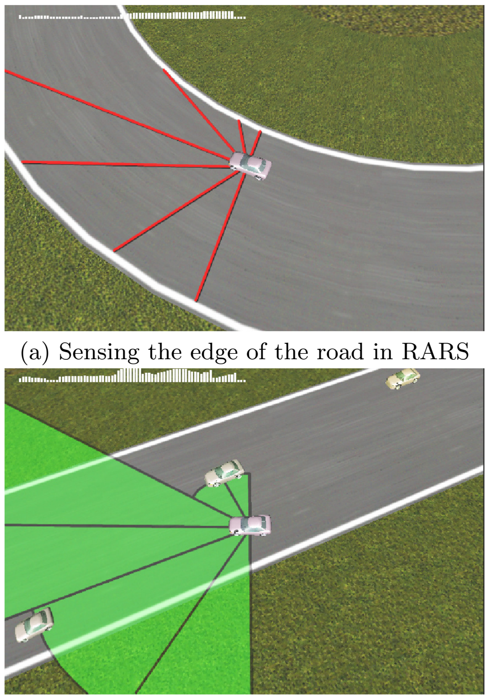

(a) (b) (c)

Figure 1: Three views of the RARS environment. In (a), a driver is shown avoiding other cars while a warning system collects data. The output of the warning system (described in Section 2.3) is shown in red in the upper-left corner of the image. In (b), a similar situation is shown, but from the perspective of the driver. (c) A 2-D overhead view of the "clkwis" track used for the experiments. All of these views represent the same environment, which can include any number of cars operated by independent controllers. Because RARS is a popular platform that accurately simulates vehicle physics and supports multiple simultaneous drivers, it makes a good simulation testbed for evolving warning systems.


(a) Sensing the edge of the road in RARS

(b) Sensing other cars in RARS

Figure 2: The two types of simulated laser rangefinder sensors. (a) Seven beams that intersect with the edge of the road give the car a sense of its position and the road's curvature. (b) To detect obstacles, the area in front of the car is divided into six sections. Multiple beams that stop when they hit obstacles are cast in each section, and the lowest becomes the value for that section. These two types of laser rangefinder input provide the warning networks with realistic egocentric input data about the road and other cars.

to fit the complexity of the problem, is designed to solve these problems. NEAT combines the usual search for the appropriate network weights with *complexification* of the network structure. It starts with simple networks and expands the search space only when beneficial, allowing it to find significantly more complex controllers than fixed-topology evolution. This approach is highly effective: NEAT outperforms other neuroevolution (NE) methods on complex control tasks like the double pole balancing task [9, 8] and the robotic strategy-learning domain [10]. These properties make NEAT an attractive method for evolving neural networks in complex tasks. In this section, the NEAT method is briefly reviewed; see [9, 8, 10] for more detailed descriptions.

NEAT is based on three key ideas. First, evolving network structure requires a flexible genetic encoding. Each genome in NEAT includes a list of *connection genes*, each of which refers to two *node genes* being connected. Each connection gene specifies the in-node, the out-node, the weight of the connection, whether or not the connection gene is expressed (an enable bit), and an *innovation number*, which allows finding corresponding genes during crossover. Mutation can change both connection weights and network structures. Connection weights are mutated in a manner similar to any NE system. Structural mutations, which allow complexity to increase, either add a new connection or a new node to the network. Through mutation, genomes of varying sizes are created, sometimes with completely different connections specified at the same positions.

Each unique gene in the population is assigned a unique innovation number, and the numbers are inherited during crossover. Innovation numbers allow NEAT to do crossover without the need for expensive topological analysis. Genomes of different organizations and sizes stay compatible throughout evolution, and the problem of matching different topologies [6] is essentially avoided.

Second, NEAT speciates the population so that individuals compete primarily within their own niches instead of with the population at large. This way, topological innovations are protected and have time to optimize their structure before they have to compete with other niches in the population. The reproduction mechanism for NEAT is *explicit fitness sharing* [3], where organisms in the same species must share the fitness of their niche, preventing any one species from taking over the population.

Third, unlike other systems that evolve network topologies and weights [4, 13], NEAT begins with a uniform population of simple networks with no hidden nodes. New structure is introduced incrementally as structural mutations occur, and the only structures that survive are those that are found to be useful through fitness evaluations. In this manner, NEAT searches through a minimal number of weight dimensions and finds the appropriate complexity level for the problem.

2.3 Training Warning Networks

Each warning network that NEAT generates receives input that describes the current state of the world. This input is normalized into the range $[0, 1]$ and is described in further detail in Sections 3 and 4. The output of each network is a *prediction* whether and when a crash is going to happen. This prediction is based on the sensor inputs over several time steps, describing the dynamics of the situation. If the predictor has a good model of the driver's behavior, it can make realistic predictions about what the driver is likely to do in the current situation, and therefore predict whether a crash is likely to happen.

Importantly, the topology of each warning network determines how many timesteps in the past it will be able to reason about. Since NEAT evolves both the topology and weights of the warning networks, it is able to determine on its own how many timesteps in the past are necessary to observe. The recurrent structures are selected during evolution based on how well they support the predictions.

The simplest kind of prediction is a binary output that predicts whether or not a crash will happen in some fixed number of timesteps. While such a system is useful, a more sophisticated prediction can be made if a network also determines *when* it expects the crash. By predicting a time, a network is in effect constantly outputting a danger level: the sooner the predicted crash, the more dangerous the situation. Such a graded warning system is likely to be more useful to human drivers, allowing e.g. different warning signals to be used depending on their urgency.

Hence, the network has a single output that is interpreted as a predicted time to crash between zero (i.e. imminent crash) and a maximum time m. In general, when the network outputs the maximum value, it means that the current situation is not dangerous. Fitness is computed as the mean squared error $\overline{E}$, accumulated while a driver drives around the track and encounters dangerous situations. Let I_t be the correct prediction at timestep t and o_t be the prediction output by the network. In the event a crash is more than m timesteps in the future, I_t is set to m. In computing $\overline{E}$, I_t and o_t, which range between zero and m, are scaled between zero and one. The mean squared error $\overline{E}$ over n timesteps is then:

$$\overline{E} = \frac{\sum_{t=1}^{n} (o_t - I_t)^2}{n}. \quad (1)$$

For the experiments in this paper, warning networks are evaluated offline from prerecorded training data from a human driver. All of the crashes in this data are hand-labeled by an experimenter. Before being used by NEAT for training, this data is parsed by a script so that each timestep is assigned the correct warning level I_t. To obtain this value for timestep t, the parser simply looks ahead to when the next crash occurs. If the next crash happens more than $t + m$ timesteps in the future, the warning level for timestep t is set to m. Otherwise, the warning level for timestep t is simply the number of timesteps until the next crash. Since each timestep of the training data is now associated with an ideal prediction, $\overline{E}$ can be computed by comparing each prediction of the warning network to this precomputed set of ideal prediction targets I_1 through I_n.

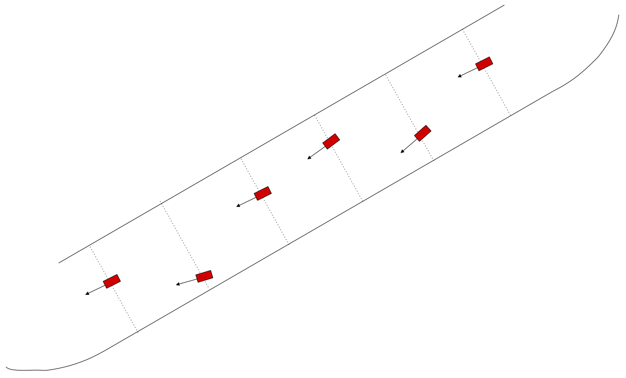


Figure 3: The initial locations of the six moving cars. Each car is evenly spaced along the backstretch of the "clkwis" track and given a small random offset and orientation. This setup provides ample opportunity for a warning system to experience moving traffic.

Using this training paradigm, it is possible to evolve networks that vary their warning level. See Appendix A for more details on the NEAT parameters used in these experiments. The following sections describe three advances that scale up this approach and might eventually allow it to be implemented in real cars.

3. WARNING WITH MOVING OBSTACLES

In previous work, NEAT was shown to be able to successfully evolve warning networks in a sparse environment, where the only collisions that could occur were with the edge of the road [7]. But how does NEAT fare in more realistic driving environments? In order to answer this question, the RARS domain was extended to include other moving cars, and warning networks were trained to predict collisions with them.

A total of six cars were added to the long straightaway of the "clkwis" track (Figure 3). They were spaced evenly along the track and were assigned a random offset from the center of the road. Each car was oriented to align with the clockwise direction of traffic, but a small offset was added to ensure that some of the cars would move across the road. All of these cars were instructed to move at a moderate constant velocity in a straight line.

In order to generate training data, a human experimenter drove a car around the track, occasionally colliding with the other cars. It proved very important for this experimenter to collect data from a variety of situations; if too few or too many crashes occurred, the networks would learn to never warn or to warn all of the time. At each timestep during this process, the rangefinder data, the velocity of the car, and any crashes were recorded. Hence, there were a total of 14 inputs: seven edge-of-road rangefinder readings, six car rangefinder readings, and the current velocity of the car. This entire data set (consisting of 16,360 timesteps) was parsed to assign target warnings to each timestep before being divided into training and testing groups (75% for training, 25% for testing). NEAT was trained and evaluated on this data for 200 generations using four-fold cross-validation.

Figure 4 depicts a network that was evolved by NEAT for this task, and figure 6 compares the mean squared error for the evolved warning networks on the test data with several fixed warning policies:

- **Always warn**: The warning system always issues the maximum warning.
- **Never warn**: The warning system never issues a warning.
- **Warn randomly**: The warning system chooses a random warning for each timestep.

The evolved networks average a low 0.028 mean squared error across the different sets of test data, performing more than twice as well as the best hand-coded policy. When NEAT was previously used to evolved networks to warn about collisions on the open road, similar performance levels were reached [7]. Subjectively, when driving a car with a joystick, the warnings appear meaningful and early enough to avoid collisions. These results suggest that despite a challenging environment with moving obstacles, NEAT is able to successfully evolve warning networks.[2]

4. LEARNING FROM VISUAL INPUT

The above results show that NEAT is able to evolve warning networks in complex environments using input from a laser rangefinder. It is difficult to work with laser rangefinders in the real world, however, because they are rather delicate and expensive. Low-resolution CCD digital cameras, on the other hand, are comparatively cheap and robust. Such cameras can also provide a much larger (albeit less precise) array of inputs, and can include color data. Given these benefits, digital cameras are an attractive source of input for a vehicle warning system.

Compared to laser rangefinder input, however, raw visual data is of much higher dimensionality and is much more difficult to interpret. For example, a processed laser rangefinder will return values that correspond to the nearest obstacles in the plane of the rangefinder with a high degree of certainty. A camera, however, can be fooled by textured and colored objects. Furthermore, a single camera image does not provide enough information to reliably extract depth information; such information must be inferred from the observed movement over several frames. Given these drawbacks, it is not clear that the learning algorithm can generate warning networks based on raw visual inputs.

To test the efficacy of evolving networks that use camera data, similar experiments to those described above in Section 3 were performed. Instead of using rangefinder sensors, however, warning networks were supplied with raw pixel data from a first-person driver's view through the windshield. The 280 pixels from a 20×14 image were converted to grayscale intensity values between 0 and 1 before being fed to an input array on the network (Figure 5).

[2]Video of these results can be found at http://nn.cs.utexas.edu/keyword?vehicle-warning

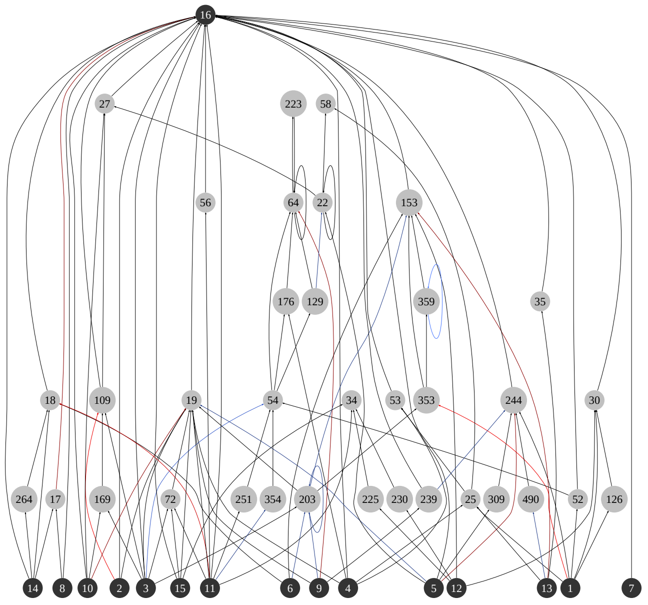


Figure 4: An example of a champion warning network evolved by NEAT after 200 generations. Weight values for the connections are depicted by color; they vary from bright red (very inhibitory) to black (neutral) to light blue (very excitatory). Input nodes are located along the bottom, the output node is located at the top, and the light gray nodes in between are evolved hidden nodes. NEAT attempts to match the complexity of the evolved networks to the complexity of the problem. For example, the recurrent connections in nodes 64, 22, 359, and 203 allow it to detect dangerous situations that are not immediately obvious from the sensor values, such as skidding off the road.

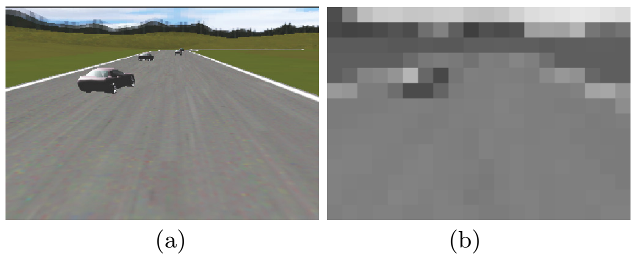


Figure 5: Training with visual inputs. (a) The original raw pixel data from a first-person camera mounted in the car and (b) a 20×14 grayscale version of the same data that is provided as input to the warning networks.

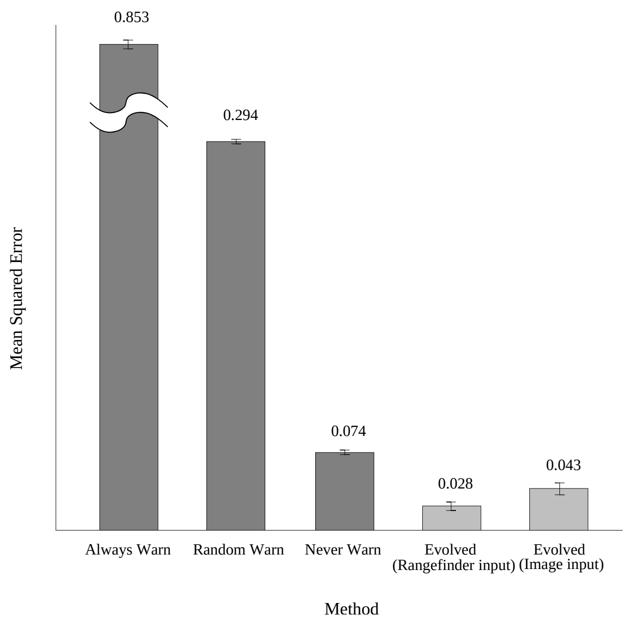


Figure 6: A comparison between warning networks evolved by NEAT and fixed baseline warning policies for warning about collisions with other cars. When networks are evolved with laser rangefinder data as input, they perform more than twice as well as the best baseline policy. Surprisingly, networks evolved with image data perform almost as well.

The results from a comparison using four-fold cross-validation are shown in Figure 6. Surprisingly, networks evolved using visual data perform almost as well as those evolved using laser rangefinder data, and still manage to significantly outperform all three baseline warning policies. This result shows that despite the high dimensionality and unprocessed nature of the visual input, NEAT is still able to generate competent warning networks.

One possible explanation for this success is that the networks are somehow taking advantage of flaws in the training and testing data. For example, if the data does not include enough variety, there may exist a single "magic pixel" that happens to correlate well with when crashes occur. Genetic algorithms are known for their ability to exploit such subtle experimental flaws.

To test the "magic pixel" hypothesis, a pixel ablation study was performed. One input of a champion warning network was ablated, i.e. forced to always show black. The performance of the network was then evaluated and the difference between the ablated performance and the normal performance was noted. The ablated input was returned to normal and another input similarly ablated, until all 280 inputs had been individually ablated and tested.

The results from this experiment are shown in Figure 7. Each pixel in Figure 7 (a) represents the performance difference when that pixel was ablated. Black pixels indicate no change in performance whereas white pixels indicate that performance degraded to the worst possible level (1.0 MSE). If the warning network was relying on a small number of pixels to generate its warnings, one would expect this analysis to reveal those pixels as being much brighter than others. As this figure shows, however, all of the pixels have roughly

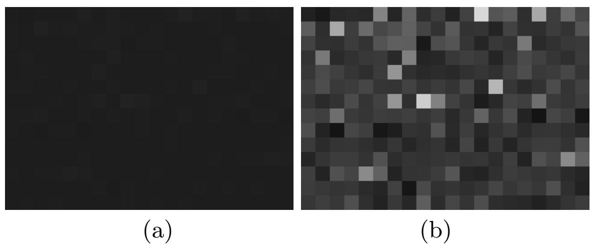


Figure 7: Results from the pixel ablation experiment. (a) Darker pixels indicate small changes in performance after ablation whereas light pixels indicate large changes in performance. (b) A normalized version of the same data. The warning network does not seem to use any single pixel to generate its warnings, but it has discovered that some areas are more informative than others.

the same color. To more clearly elucidate this result, Figure 7 (b) shows the same analysis but with the values of the pixels normalized. There are many pixels in this figure with a light color, which suggests that the warning network is not relying on a single pixel to generate its warnings.

Interestingly enough, some of the many pixels that seem to contribute are located at the very top of the image, where the rendering of the sky almost always yields a constant color. It is possible that the warning network is using such constant pixels as bias values, despite being provided with a bias neuron that is normally used with neural networks. Although evolution is not taking advantage of any single "magic pixels", it does seem to be exploiting the regularity of the sky in the simulated world. The next section explores the question of whether or not this approach still work in the real world, where the environment lacks the uniformity of simulation.

5. LEARNING ON A ROBOTIC VEHICLE

The above approaches, developed in simulation, were transferred to the real-world and implemented on a robotic vehicle testbed. In a manner similar to the simulation, the robot was trained to warn about impending collisions and lane departures while moving through a cluttered environment.

The robot used for this experiment was a customized Applied AI GAIA robot with a SICK laser rangefinder and a Bumblebee stereo camera (Figure 8). Both the rangefinder and camera were configured to save 10 frames of data per second to the local storage device of the robot. The rangefinder generated 180 measurements in one-degree increments, but these values were averaged over 20 slices to produce 20 measurements over the 180 degree span. The stereo data from the camera was discarded and the resulting single images were compressed down to a 20×14 pixel size.

The robot was piloted through a mock roadway littered with stationary black obstacles with a height similar to that of the robot. A total of 8094 frames of training data were recorded, including 112 collision events. This data was processed offline and fed into NEAT to generate warning networks. In this preliminary experiment, all of the available data was used for training; the results in Figure 9 show two examples of performance on this training set.

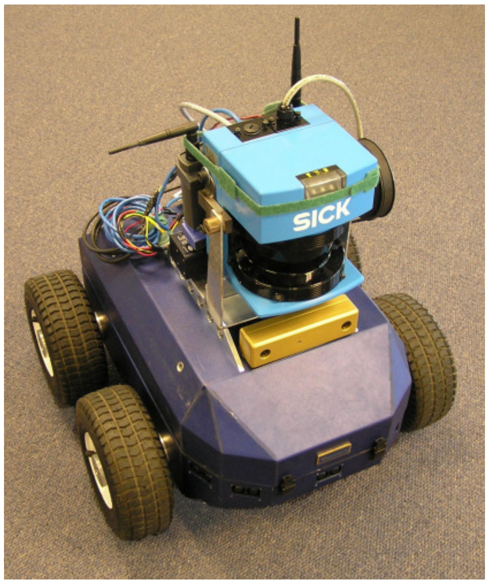


Figure 8: The robotic vehicle (an Applied AI GAIA) used for real-world training, with its SICK LMS-200 laser rangefinder and Point Grey Bumblebee digital stereo camera.

In Figure 9, each example shows the events leading up to a collision with a stationary object at one-second intervals. Thus the first image (on the left for each example) depicts the robot three seconds before impact, whereas the last image (on the right for each example) shows the robot at the moment of impact. A history of the warnings generated by a warning network is shown along the top of each image. Rangefinder input is shown in the top set of images. In both examples, as the object draws closer, the warning levels increase appropriately.

These results suggest that NEAT does not rely on the regularity of the simulated world to generate robust warning networks. Despite having to deal with noisy laser rangefinder data and a cluttered visual environment, NEAT is able to successfully evolve warning networks in the real world.

6. CONCLUSIONS

This paper presents three advances in the application of NEAT to a vehicle collision warning domain. First, NEAT was evaluated in a complex, dynamic environment that included other cars; it outperformed three hand-coded strawman warning policies and generated warning levels consistent with those of an open-road warning system. Second, the processed laser rangefinder input to the warning networks was replaced with raw pixel data from a simulated camera. Surprisingly, NEAT was still able to generate warning networks that outperformed the baseline hand-coded warning policies. Finally, this approach was evaluated using a real-world robotic vehicle testbed. Despite noisy real-world sensor data, preliminary training experiments suggested that NEAT was able to generate robust warning networks using both laser rangefinder and visual sensors. These three results set the stage for developing warning networks for real-world traffic, which may someday save lives in real vehicles.

7. FUTURE WORK

One obvious next step for this work is to implement this system in a full-size vehicle in an outdoor environment. It will be important in particular to test how well the networks trained with visual data react to changing lighting conditions. Such an extension may require extending the visual field significantly.

Another interesting avenue for future work is online training, which has the potential to allow warning networks to become customized to individual drivers [7]. Extensions to the NEAT algorithm like FS-NEAT [12], which allows selecting the most appropriate input features as part of the evolution, may also turn out useful for scaling up to real cars.

Finally, it could be very useful to continue to compare how different types of input affect the learning process. The work in this paper shows that NEAT can learn with either low-level pixel data or medium-level rangefinder data. There are many current visual processing systems that provide different high-level information from a visual stream of data. Such high-level input is attractive because it offers the possibility of greatly simplifying the learning process. On the other hand, high-level input is by definition fairly abstract, and may not always provide the right information for every dangerous situation. Understanding which level of input works most effectively with evolution is an interesting direction for future work.

8. ACKNOWLEDGMENTS

The authors would like to thank all of the programmers that contributed to the open-source RARS project, which is available at http://rars.sourceforge.net. Thanks to Robert Vogt for providing valuable technical assistance during this project. This research was supported in part by a grant from Toyota, USA.

APPENDIX A.

This appendix describes the NEAT parameters used for experiments in this paper. For more information about these parameters, see Stanley and Miikkulainen [9]. The coefficients for measuring compatibility were $c_1 = 1.0$, $c_2 = 1.0$, and $c_3 = 3.0$. The survival threshold was 0.2. The champion of each species with more than five networks was copied into the next generation unchanged. The weight mutation power was 0.06. The interspecies mating rate was 0.05. The number of target species was 5. The drop-off age was 1000. The probability of adding recurrent links was 0.2. Small variations of these values yield approximately equivalent results.

9. REFERENCES

[1] N. H. T. S. Administration. Traffic safety facts: A compilation of motor vehicle crash data from the fatality analysis reporting system and the general estimates system, early edition. Technical report, National Center for Statistics and Analysis, U.S. Department of Transportation, 2004.

[2] E. Dickmanns and B. Mysliwetz. Recursive 3-D road and relative ego-state recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):199–213, 1992.

[3] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In

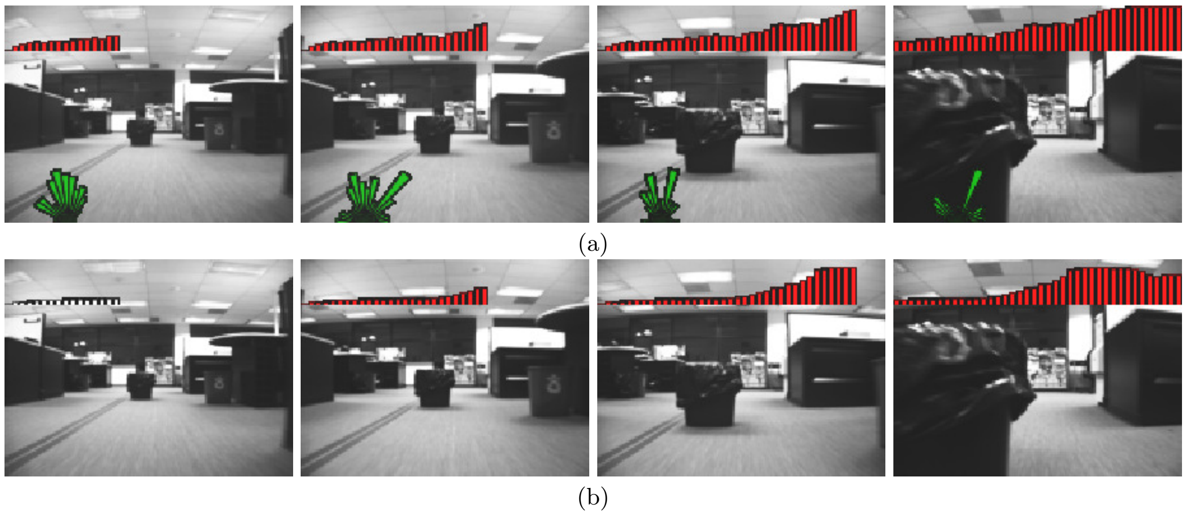


Figure 9: An example robot-generated warnings for a single crash (a) using laser rangefinder data as input and (b) using digital camera data as input. The 20 rangefinder values are shown in green in the bottom-left corner of the images in (a), and the warning levels generated by the evolved network are shown in red in the top-left corner of the image. Despite noise in the rangefinder data and visual clutter in the visual data, the results from this training suggests that the networks have learned to successfully predict collisions.

Proceedings of the Second International Conference on Genetic Algorithms, pages 148–154, 1987.

[4] F. Gruau, D. Whitley, and L. Pyeatt. A comparison between cellular encoding and direct encoding for genetic neural networks. In J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pages 81–89. MIT Press, 1996.

[5] M. Haag and H.-H. Nagel. Combination of edge element and optical flow estimates for 3d-model-based vehicle tracking in traffic image sequences. *International Journal of Computer Vision*, 35(3):295–319, 1999.

[6] N. J. Radcliffe. Genetic set recombination and its application to neural network topology optimization. *Neural Computing and Applications*, 1(1):67–90, 1993.

[7] K. Stanley, N. Kohl, R. Sherony, and R. Miikkulainen. Neuroevolution of an automobile crash warning system. In *Proceedings of the Genetic and Evolutionary Computation Conference 2005*, pages 1977–1984, 2005.

[8] K. O. Stanley and R. Miikkulainen. Efficient reinforcement learning through evolving neural network topologies. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)*, 2002.

[9] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2), 2002.

[10] K. O. Stanley and R. Miikkulainen. Competitive coevolution through evolutionary complexification. 21:63–100, 2004.

[11] F. Thomanek, E. D. Dickmanns, and D. Dickmanns. Multiple object recognition and scene interpretation for autonomous road vehicle guidance. In *Intelligent Vehicles Symposium*, 1994.

[12] S. Whiteson, P. Stone, K. O. Stanley, R. Miikkulainen, and N. Kohl. Automatic feature selection in neuroevolution. In *GECCO 2005: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1225–1232, June 2005.

[13] X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.