Vision Based Intersection Navigation

Todd M. Jochem, Dean A Pomerleau, and Charles E. Thorpe
The Robotics Institute,
Carnegie Mellon University, Pittsburgh, PA 15213
Phone +01-412-268-3260; Fax +01-412-268-5571; email contact: tjochem@ri.cmu.edu

Abstract

Much progress has been made toward understanding the autonomous on-road navigation problem using vision based methods. A next step in this evolution is the intelligent detection and traversal of road junctions and intersections. The techniques presented in this paper are based on a data driven, active philosophy of vision based intersection navigation. Traversal is accomplished by imaging relevant parts of the intersection using a combination of active camera control techniques and a Virtual Active Vision tool called a virtual camera. By monitoring the response of the underlying lane keeping system to the created images, intersections and road junctions can be detected and traversed.

Introduction

Most current vision based intersection navigation systems require some a-priori model of the intersection geometry or markings. The system presented in this paper can detect and navigate through intersections of arbitrary geometry and unknown location using only visual cues. If geometric information is available, it can be incorporated to more quickly define the probable location of the intersection by constraining the search for intersection branches.

The core of the intersection navigation algorithms presented in this paper is the ALVINN lane keeping system enhanced with virtual cameras and active camera control [1]. Using these tools, ALVINN can detect intersections from both stationary and moving vehicles. The detection and traversal algorithms are completely image driven - they need no other positioning sensor for detection and navigation to be successful. The system detects the presence of an intersection by looking for the intersection branches. When an intersection branch is present, a virtual camera will image it so that ALVINN produces a high confidence value. After all branches have been found, the system can use high level information to choose the appropriate branch and begin navigating through the intersection.

After the intersection has been detected and the branch to be navigated onto has been selected, traversal of the intersection begins. During traversal, the system continually updates the location and orientation of the vehicle with respect to the branch using output from the ALVINN system. In addition, this same information is used to update the position of the virtual camera with respect to the vehicle so that it continues to image the branch appropriately. When the system detects that the road branch that the vehicle is moving onto is about to move out of the field of view of the actual camera, it pans the actual camera appropriately. In this manner, only visual information is needed to successfully traverse the intersection.

Results are presented for two different intersection navigation scenarios. In the first scenario, the system has a-priori knowledge that specifies the approximate location of an upcoming road branch, along with the branch geometry and road type. Because searching is not needed, the system can detect the branch while the vehicle is in motion and smoothly navigate onto it. In the second scenario, the system only has knowledge that an intersection is present in front of the vehicle. The system does not know the orientation of road branches that are intersecting at this road junction. The goal is to locate each intersection branch and navigate onto one of them. Results are presented from live vehicle runs which show the performance of the system in both scenarios.

Experimental Description

After the intersection branches have been detected, traversal can begin [2]. A robust traversal algorithm must overcome two potential problems: a fixed camera location and violations of the assumptions about the geometry of the road branch. The fixed camera problem, which limits the effective field of view of the system, was overcome by simply placing the camera on a pan-tilt mount located on the roof of the vehicle. The geometric constraint violation problem, which caused poor traversal performance in prior experiments [2], was resolved by using image-derived information to continually update the estimate of the branch position. Adding these capabilities

0-7803-3652-6/96/$5.00 © IEEE

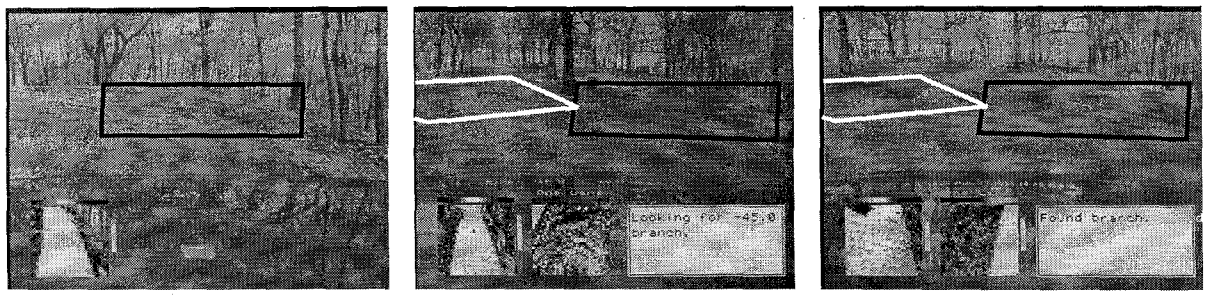

Image 1 Image 2 Image 3

Figure 1 Intersection branch detection from a moving vehicle.

allowed the system to reliably detect and navigate two test intersections. The first was a "Y" intersection while the other was a "T" junction from a driveway onto a rural road. While not exhaustive, these two locations are typical of intersection geometries encountered in everyday driving.

Known Geometry and Unknown Location

In this experiment, the goal was to move along a single lane road, search for and detect a branch of "Y" intersection, and drive onto it. In order to accomplish this, the moveable camera was required. This camera could be positioned so that the road, as well as a substantial portion of the anticipated road branch, could be imaged. If a fixed camera had been used, the number of branch locations and the amount of actual and virtual camera view overlap which was possible would have been limited.

Initially in this scenario, ALVINN controlled the vehicle in normal lane keeping mode. See Image 1 of Figure 1. While driving, the system received a message that an intersection was approaching. Although the exact location of the branch was not known, its orientation with respect to the current road segment was given. Using this information, the system created the appropriate virtual camera view, called the Detection View, which would properly image the branch when it appeared. For this experiment, the target road branch was oriented approximately 40 degrees left of straight ahead. In addition to being angled 40 degrees, the Detection View was typically located 7 meters in front of the vehicle. This distance was selected so that the branch would be detected enough in advance to perform the traversal maneuver but close enough so that violations in the flat world assumption would not become significant.

After creating the Detection View, the system determined if the current pan location of the actual camera was sufficient to image both views completely. If not, which was typically the case, the system automatically panned the camera so that the largest portion of the Detection View was in the field of view of the actual camera, while maintaining the entire Driving View in the actual camera's field of view. See Image 2 of Figure 1. Note that after the actual camera had been panned, it is no longer in the same orientation as when the ALVINN network was trained. But because the virtual camera is at a fixed location with respect to the vehicle and is independent of the actual camera location, the images it created allowed ALVINN to continue driving reliably.

New images from the Detection View were created approximately 4 times per second and passed to ALVINN's neural network for processing. The intersection branch was considered detected when the IRRE confidence value of the network, in response to a Detection View image, became greater than a predetermined threshold value. See Image 3 of Figure 1. The threshold value was typically set to 0.75. At this point, the system began to localize the intersection branch and navigate through the intersection. This process is described in detail in later sections.

Unknown Geometry and Known Location

This intersection detection scenario is the opposite of the previous. In this case, the location of the intersection was known, but the geometry of the intersection was not. Specifically, ALVINN had knowledge about where the center of the intersection was located with respect to the vehicle, but did not know the orientation of any of the intersection's branches. The goal was to find each branch and store it location for further processing.

The branch detection process began with the vehicle located a known distance from the intersection center. The detection algorithm uses a radial search technique to create virtual camera views which image different hypothesis branch locations. Virtual views are created a fixed distance from the intersection center at varying orientations. For this experiment, the angular change between hypothesis views was 45 degrees, while the vehicle's distance

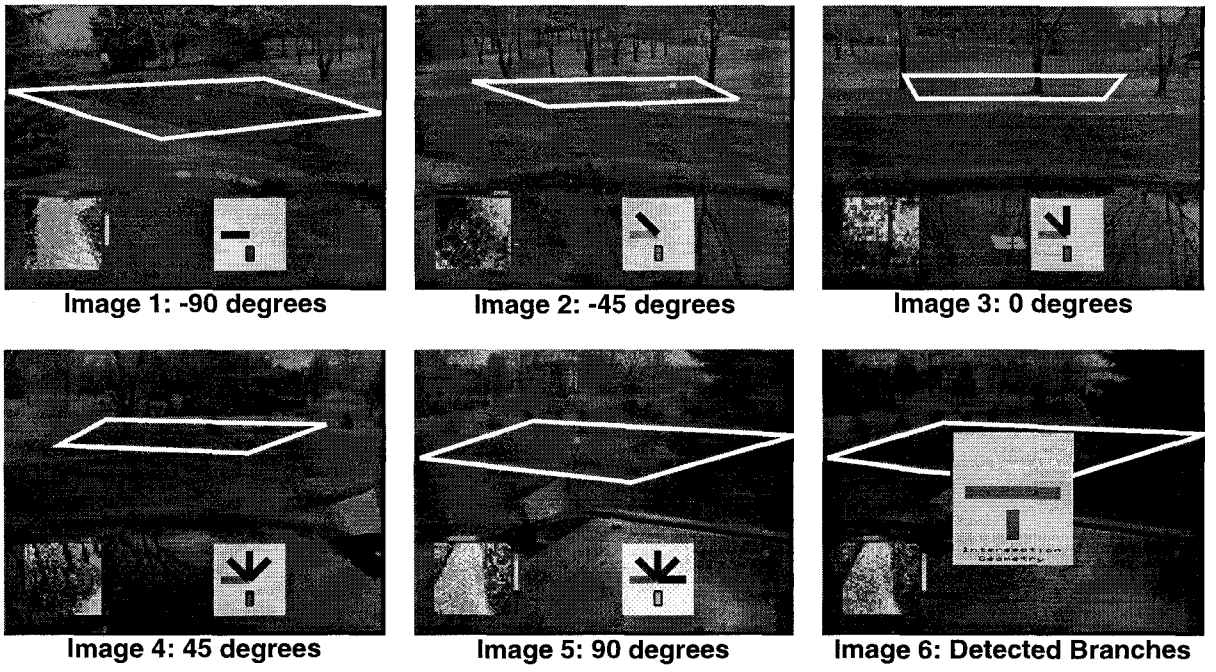

Figure 2 Searching for "T" intersection branches.

from the intersection center ranged between 7 and 10 meters. Images taken at each of these hypothesis branch locations are shown for the "T" intersection in Figure 2. Image 6 of Figure 2 shows which hypothesis intersection branches the system believes are likely to be actual branches as determined by the simple detection method described in the next paragraph.

The basis for signaling detection is a high IRRE value. If the hypothesis view images an actual road branch, the corresponding IRRE confidence metric will be high, and the orientation of the branch being examined can be saved for further processing. For preprocessed images created from virtual cameras which did not image actual road branches in Figure 2, the IRRE value is very low, indicated by the short bar next to the preprocessed image in each of the images. But for the images which were of actual road branches, the confidence value is significantly higher. From this examination, it is evident that by thresholding based on the IRRE value, hypothesis views which image actual road branches can be discriminated from those that do not.

Navigation Using Active Camera Control

Before traversal takes place, the road branch must be localized to a greater degree of accuracy than was done for detection. This is necessary because of ALVINN's ability to correctly respond to images in which the vehicle appears misaligned with the road. Because of this, the exact location and orientation of the road branch with respect to the vehicle is not precisely known. Because lateral translation and orientation errors in virtual view alignment cannot be determined from a single image and its associated output, the following two step branch localization process is used to refine the road branch location estimate.

The first step in localizing the branch further is to use the output displacement of the network to update the position of the virtual camera imaging the road branch. This is done by moving the view laterally, perpendicular to the hypothesis branch direction, for a distance equal to the output displacement of the network. After the view has been moved, an image is created from this new location and passed through ALVINN's network, producing another output displacement which is again used to adjust the view. This process is repeated until the output displacement of the network changes sign, meaning that the current and last view have "bracketed" the view location which will produce zero output displacement. In this last step, the final displacement from straight ahead is very small.

Although the first phase of road branch localization causes the output displacement of the network to become nearly zero, the orientation of the view with respect to the road branch cannot be assumed to be correct. Figure 3 illustrates this concept. In this figure, the preprocessed image along with ALVINN's output displacement from each image, when the vehicle is in both the left and right configurations with respect to the road, is shown. Note that in each case the displacement is near zero although the vehicle is only

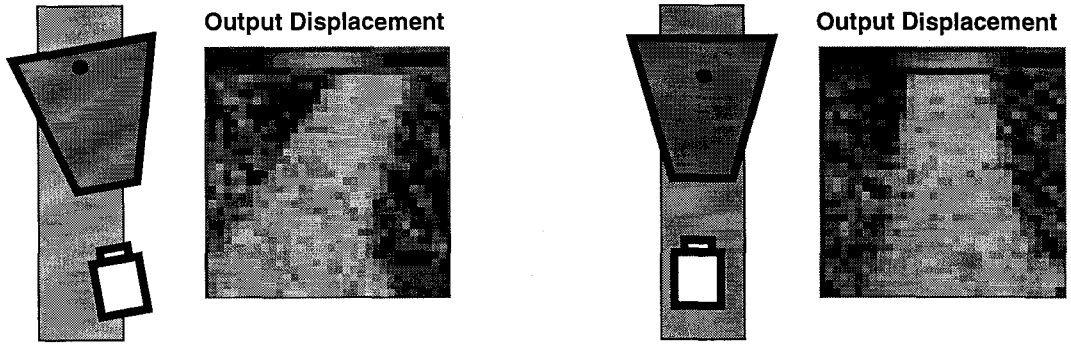

Figure 3 Possible alignments with zero output displacement.

properly aligned with the road in the right example. This occurs because ALVINN is trained to produce the necessary displacement to return the vehicle to the center of the road the lookahead distance in front of the vehicle. In both cases, the required displacement to return the vehicle to the center of the road is near zero.

In order to accurately determine the intersection branch orientation, a second view is required. This view, called the Projection View, is typically created between 3 and 5 meters in the direction of the current estimated road branch orientation. Figure 4 shows the actual road scene along with a diagram of the original and Projection View arrangement.

If the original view is properly aligned with the intersection branch, creating an image using the Projection View and passing it through ALVINN's network should yield an output displacement close to zero. This is because the previous alignment step reduced the lateral offset of the original view to near zero, in effect centering the original view over the longitudinal axis of the road branch. If the original view is at the correct orientation, projecting 5 meters along the branch orientation should also create a view which is centered over the longitudinal axis of the intersection branch. As is shown in Figure 4, this is not usually the case. Although the original view has zero displacement, indicated by the centered gaussian hump of activation over the preprocessed image, it is not aligned correctly with the intersection branch. Because the original view was misaligned, the Projection View is also misaligned, which results in a non-zero output displacement. In Figure 4, the gaussian hump indicating the network output displacement created from the Projection View image is shifted right to reflect this misalignment.

The output displacement difference from zero that the Projection View image produces is a measure of the misalignment in orientation between the original view and the intersection branch. By using the output displacement from the Projection View, the projection distance, and the location of the original and Projection Views, the amount of this angular misalignment can be computed. After rotating the original view to its correct orientation, its lateral position must also be corrected. This is necessary because the rotation correction was done about the original view's location and not about a point on the longitudinal axis of the intersection branch. From the same information used to compute the orientation error, the lateral offset error, which is a result of the orientation error correction, can also be computed [1]. After both error values have been computed, they are used to update the original view location and orientation so that it more closely matches the intersection branch geometry. Once this step of localization is finished, traversal of the intersection can begin.

Traversal

There are two issues which must be considered and resolved in order for intersection traversal to be successful: tracking the branch as the vehicle moves through the intersection and computing the correct steering arc to execute.

The branch tracking problem was solved by adapting the branch localization algorithm presented in the previous section. During traversal, the system continually updated the location and orientation of the original virtual camera view by repeating the alignment procedure presented in the previous section. In addition, when the original view was about to move out of the field of view of the actual camera, the system automatically panned the actual camera appropriately. The ability of the system to correctly orient and localize the intersection branch during traversal is shown for the "T" intersection in Figure 5. Note that a camera pan occurs before each image in this figure.

Creating an acceptable vehicle control algorithm for navigating intersections was one of the most difficult tasks in this work. The majority of the

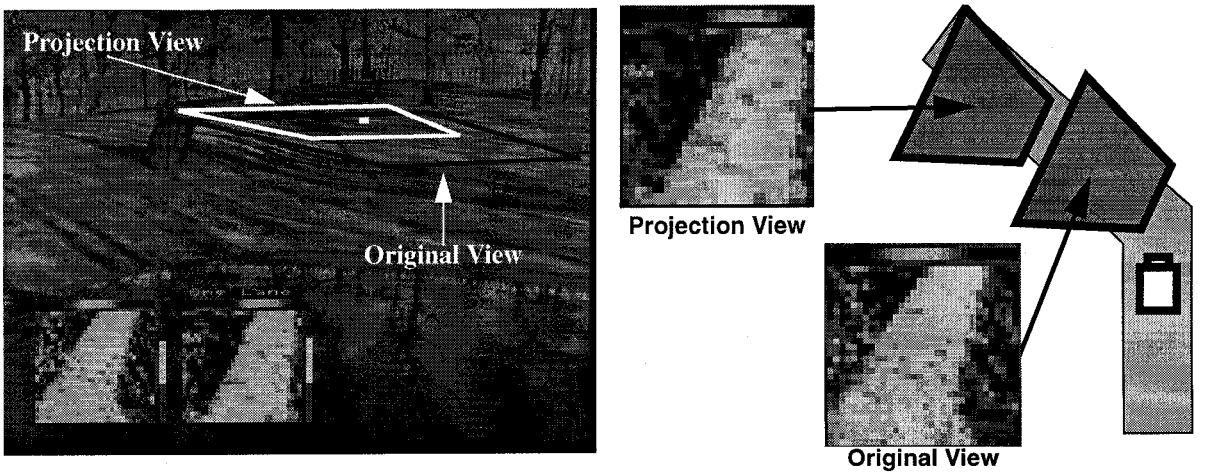

Figure 4 Orientation localization.

methods tried caused the vehicle to either severely cut corners, overshoot, or generally become misaligned with the road. A contributing factor to these problems was the lack of accurate geometric information about the intersection branch as the vehicle turned.

This problem was alleviated by tracking the intersection branch using a combination of traditional and virtual active vision techniques. But given this, many of the vehicle control algorithms still had difficulty matching the vehicle heading to the road orientation. Based on this observation, the vehicle control algorithm shown in Figure 6 was developed. This algorithm takes into account the branch orientation as determined by the localization algorithm.

The vehicle control algorithm finds tangent points on two lines representing the vehicle's current heading and the intersection branch orientation. The first tangent point, P1, is defined to be the current vehicle position while the second point, P2, is on the intersection branch axis. The distance along the branch that P2 is located, measured from the intersection center point, C, is defined to be equal to the distance from P1 to C. After being computed, this distance is held fixed throughout the intersection traversal. C is computed before traversal begins by finding the intersection point of the branch axis and the line representing the vehicle heading. As mentioned earlier, the orientation of the branch axis is not fixed at the hypothesis view orientation, but rather is the refined branch orientation derived during the branch localization phase.

This construction insures that a circle can be found which will intersect P1 and P2 tangentially. The radius of the circle that intersects these tangent points is the arc that the vehicle uses to drive through the intersection. For each new image, the tangent point, P2, and arc to drive are recalculated based on the new location of the intersection branch so that any errors in vehicle control, positioning, pan angle, or nonlinear branch geometries are taken into account.

Results and Discussion

Using simple thresholding as the discriminating technique, the system was able to successfully detect each intersection branch in 33 of the 35 cases on the "Y" and "T" intersections. These trials were distributed about evenly over the moving vehicle "Y," the stationary "Y" and "T" detection scenarios. In no cases did the system detect a branch which was not present.

Both failure cases were in the "T" scenario. In one case, the system successfully detected one of the two branches. The other branch's confidence value fell just below the threshold but was still much higher than any of the other three hypothesis locations. In the other failure case, neither branch was detected. For this case, of the two real branches that should have been detected, one did have a noticeably higher IRRE value, but it was still below the detection threshold. The other branch's IRRE value was not significantly different than any of the other branches. In this case the problem can be attributed to a change in the ambient lighting, from overcast skies to sunshine, to which the camera was not able to properly adjust. In any case though, this indicates that although detection is robust, it is not foolproof and redundant branch verification procedures are necessary.

In all 34 cases which the system detected at least one branch, it was able to properly move the vehicle onto the branch and continue driving. The system was able to drive the vehicle onto the left and right branches of the "T" intersection as well as navigate onto the 45 degree branch of the "Y" intersection.

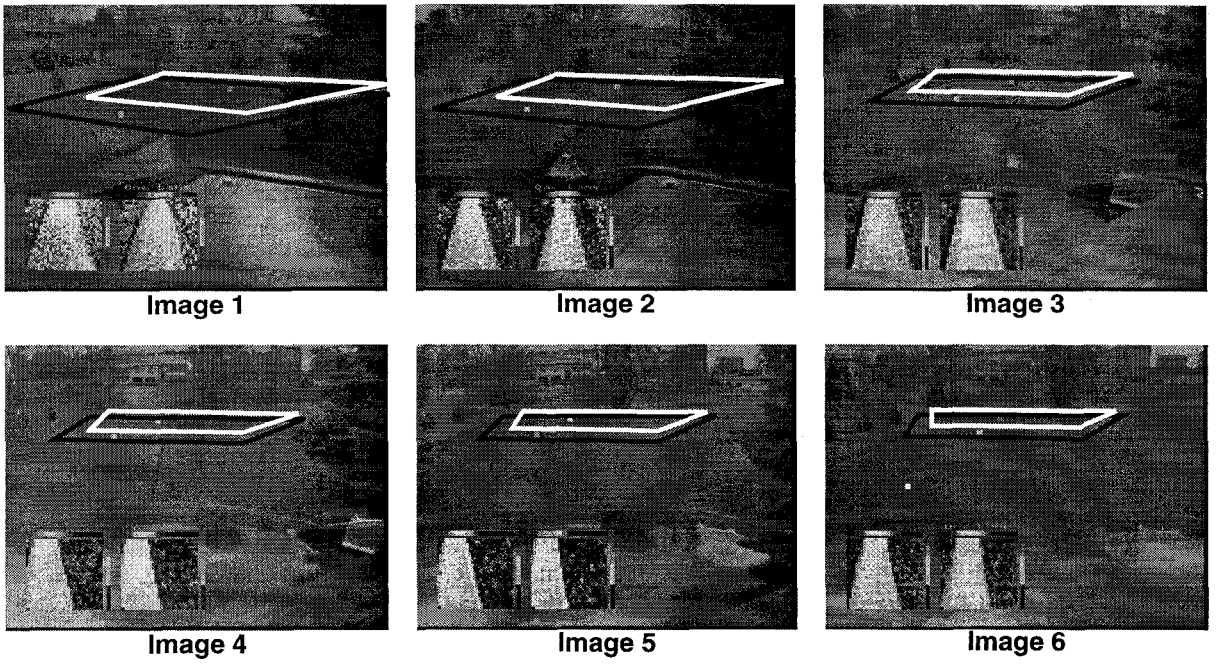

Figure 5 "T" Intersection traversal.

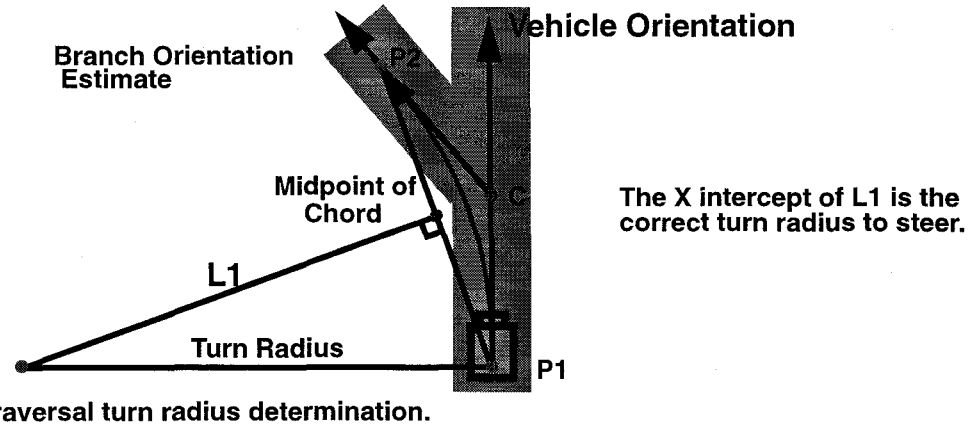

Figure 6 Traversal turn radius determination.

Although the control algorithm during traversal is very simple, having a moving camera and tracking the road branch throughout the maneuver allowed it to work reliably over the experimental domain.

It is reasonable to assume that the detection method will work for any road branch type which the base neural network can learn to drive on. If this assumption is true, this system will have an advantage over other road and intersection detection systems which require the researcher to program in new detection methods when new road types are encountered.

Acknowledgments

Sponsorship for this work was provided by the USDOT under contracts DTNH22-93-C-07023 and DTFH61-94-X-00001. The authors would also like to thank Delco Electronics for providing the testbed vehicle on which much of this work was conducted.

References

[1] Jochem, T. and Pomerleau D. "Life in the Fast Lane: The Evolution of an Adaptive Vehicle Control System," AI Magazine, Volume 17, No. 2, pp. 11-50, Summer 1996.

[2] Jochem, T., Pomerleau, D., and Thorpe, C. "Vision-Based Neural Network Road and Intersection Detection and Traversal," IROS 95, Pittsburgh, PA, USA.