A Hardware Suitable Integrated Neural System for Autonomous Vehicles – Road Structuring and Path Tracking

Udhay Ravishankar and Milos Manic

***Abstract*—Current developments in autonomous vehicle systems typically consider solutions to single problems like road detection, road following and object recognition individually. The integration of these individual systems into a single package becomes difficult because they are less compatible. This paper introduces a generic Integrated Neural System for Autonomous Vehicles (INSAV) package solution with processing blocks that are compatible with each other and are also suitable for hardware implementation. The generic INSAV is designed to account for important problems such as road detection, road structure learning, path tracking and obstacle detection. The paper begins the design of the generic INSAV by building its two most important blocks: the Road Structuring and Path Tracking Blocks. The obtained results from implementing the two blocks demonstrate an average of 92% accuracy of segmenting the road from a given image frame and path tracking of straight roads for stable motion and obstacle detection.**

I. INTRODUCTION

AUTONOMOUS Vehicular Navigation is a broad field of research aimed at developing unmanned vehicle maneuvering. By the definition of autonomous, the system must learn its apparent surroundings without the supervision of the passengers. A variety of autonomous navigation techniques exist involving different types of vision sensors. Dickmanns in [1] investigates the development of such machine vision sensors for road vehicles. The paper concludes that the camera type sensor is the most suited vision for vehicular systems as it provides lot more information about the surroundings. The challenge then lies in processing this load of information into useful information that aids the autonomous vehicle in making critical decisions. Examples of work done with this type of vision, so far, are in [2]-[13].The problem primarily involves road recognition, path tracking and obstacle detection. There are, however, other types of problems associated with camera sensors alone and these are highlighted in [14]. To accomplish a system with the capability of learning and adapting autonomously, neural network approaches are generally utilized.

Neural networks or neural computing involve algorithms that enable the system to train, learn and adapt according to its environment. The algorithms are generally classified as supervised or unsupervised learning. Supervised learning algorithms implement a teacher-student model, i.e. learning based on known outcomes, while unsupervised learning algorithms implement a researching model, i.e. outcomes are unknown and are yet to be discovered. Examples of supervised learning algorithms are the Error Back Propagation, Bisecting, Least-Mean-Square, etc. Unsupervised learning algorithms are typically data mining algorithms.

Neural networks have been implemented in the field of autonomous vehicles [4]-[8] and in other related fields such as Driver Assistance Systems [9]-[11] and Mobile Robots [15]-[20]. In autonomous vehicles, solutions to only individual problems such as road detection, road following, obstacle detection and motion detection were presented. Leibe et al, in [7], and Bertozzi et al, in [13], however couple two issues in their solution. Some of them, such as in [12] and [13], don't use neural networks.

Integrated solutions have instead been implemented in Driver Assistance Systems (DAS), [9] and [11]. Here, the assumption is that there exists a human driver to maneuver the vehicle. Therefore, although these systems perform some autonomous processing, they rely only on certain issues that are of importance in driver assistance, e.g. analyzing critical motions of nearby vehicles [9].

In an autonomous driving scenario in the real world, the vehicle is subjected to uncertainty in varying conditions, conditions that are also unknown during the vehicular system design phase. These issues can be overcome by designing a system that is trained to do certain things like road recognition, path tracking, obstacle detection, etc, in one package. Also, since the human control is eliminated in this type of system, the system should instead mimic human responses.

In humans, there exists a high level of computational parallelism when analyzing the surroundings. In electronics, computational parallelism can be realized only in the hardware design. The only problem with parallelism is that it introduces bulkiness. For a camera sensor, this bulkiness will depend directly on the camera's resolution. Hence algorithms must be more simplistic to reduce the bulkiness to some extent. The algorithms developed in this paper have a simplistic approach in solving the problems assuming that there exists such parallelism.

This paper introduces a generic INSAV that aims to solve multiple problems, such as road detection, road structure learning, path tracking and obstacle detection, in one package. The INSAV has processing blocks that are compatible with other blocks thereby simplifying the tasks of each block. In the attempt to build the generic INSAV, the paper then focuses on the design of its two most important blocks, the Road Structuring and Path Tracking Blocks. Each of these blocks combine the attributes of neural computing and hardware implementation.

Since the type of sensor used is the digital color camera, the Canny Edge Detection Algorithm (CEDA) is used to structure the focused scene. The CEDA was introduced by Canny in

1986 to solve the weaknesses of earlier algorithms in detecting weaker edges which are typical in outdoor scene images [21]. Hence the CEDA was selected because of its robustness on outdoor scene images. The structure of the focused scene aids the Road Structuring Block which in turn aids the Path Tracking Block.

The rest of the paper goes as follows: Section II elaborates on the generic INSAV design followed by the elaboration of the Road Structuring and Path Tracking Blocks in Section III. Section IV shows the obtained results from experimentation. The paper finally concludes in Section V with future work.

II. The Integrated Neural System for Autonomous Vehicles (INSAV)

The system is designed based on the Psychophysical Evidences highlighted in [9]. These systems involve Hierarchical Processing, Neural Configurability, Adaptive Response and Attention Models.

Hierarchical Processing allows the system to have different levels of analysis and processing. These levels typically include Sensory, Perceptual, Syntactic and Semantic analyzers. Neural Configurability, enables the system to use previously learned or stored information to understand its current environment. To implement this, a memory block is required from where it can retrieve or store information. Adaptive Response provides perceptual ability for the system. It involves structural analysis and motion analysis working interactively together. For vehicular systems, the Adaptive Response enables the vehicle to perceive any forthcoming obstacles ahead in time. Lastly, Attention Models guide the system to focus on anticipated areas for faster responsivity (not considered in the INSAV).

The INSAV system is shown in Fig.1. The designed INSAV, like the one in [9], utilizes three analyzers namely the Sensory, Perceptual and Conceptual Analyzers. The Sensory Analyzer consists of an Edge Detection Block and a Road Structuring Block (RSB). Since the vehicle must know the road at all times, the RSB has its place in the Sensory Analyzer. Its main goal is the provide the INSAV information regarding the current road structure.

The Perceptual Analyzer implements the Adaptive Response feature mentioned earlier. It consists of a Path Tracking Block (PTB), a Motion Classifier Block and an Obstacle Detection Block. The latter two blocks depend on a Motion Computation Block. The PTB takes the road boundary points as inputs from the RSB. Its main goal is to locate the Vanishing Point (VP) on the image. This VP aids in classifying the motions of the objects in the scene in terms of hazardous and non-hazardous motions. From this classification, the system will be able to perceive obstacles in the near future and adapt accordingly.

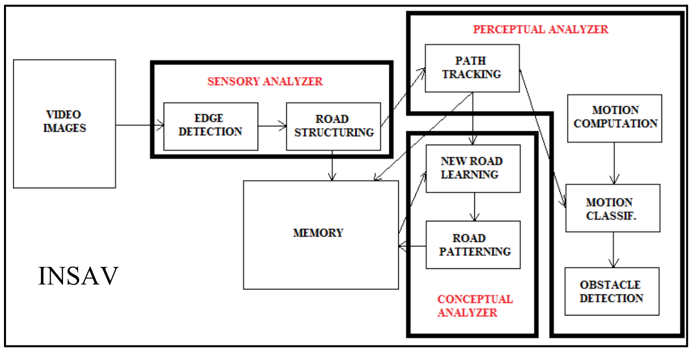

Fig. 1: The presented INSAV

Assuming a vehicle is moving along a straight road, all other vehicles, moving on the same road, tend to move along the VP, either away from it or towards it. The perceived expansion and contraction of these moving vehicles, too, follow along the VP. Therefore, once the system identifies the VP, it can create line segments on the image, to map the motions of the objects in sight. Any deviations from these line segments can alert the system's attention on those motions. The line segments also help in understanding the nature of object motion in the scene from which it can identify highly probable obstacles.

The Conceptual Analyzer consists of blocks that enable the system to learn new road structures and create road patterns that tell the system about how the road is bending or about any approaching cross-road or T-junction. This ability allows the system to prepare for making the required turns.

As mentioned earlier, the PTB takes inputs from the RSB. Therefore, like the eye that has cells that do structure analysis and motion analysis in parallel but interactively at a higher level [22], [23], the designed system too, performs road structuring and motion computation in parallel and then classifies the motions based on the VP obtained from PTB.

III. The Road Structuring & Path Tracking Neural Algorithms

This section elaborates on the algorithms implemented for the two most important blocks of the INSAV, i.e. the RSB and PTB.

A. *Neural Road Detection and Training (Road Structuring)*

This block assumes that the Memory Block of the INSAV contains data pre-mined on road colors into a road color sphere in the RGB space during, say, a factory training stage. Hence, this algorithm utilizes a supervised learning approach.

The Canny Edge Detection Algorithm (CEDA), along with the frame difference image to enhance the edge detection, initially structures the scene provided in the training image frame, $\boldsymbol{I}$. The RSB then initiates a training stage during which a road color sphere, $\boldsymbol{S}$, shown in Fig. 2 is provided. This road sphere, $\boldsymbol{S}$, represents a single type of road color. The sphere, $\boldsymbol{S}$, has a center-of-gravity (COG), COG_S, and a radius. In the training stage, the pixel colors on $\boldsymbol{I}$, were matched with $\boldsymbol{S}$. Then the identified road region set, $\boldsymbol{R_{reg}}$, defined as the set of pixel locations whose colors matched $\boldsymbol{S}$, is given as,

$$R_{reg} = \{(i,j) \,\forall\, I_{ij}(R,G,B) \in S\} \qquad (1)$$

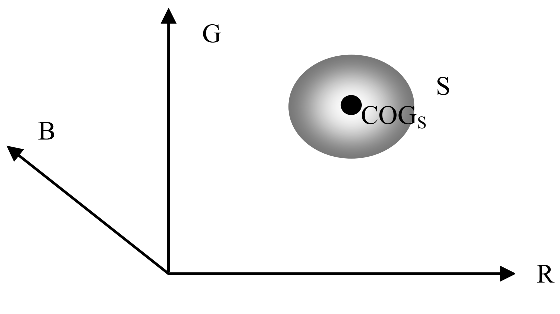

Fig. 2: Representation of the road sphere S.

where (i,j) are the pixels locations on ***I***. The set $\boldsymbol{R_{reg}}$ also had pixels that were not the road. Assuming the pixels are neurons known as Pixel Neurons (PN), processed in parallel, with inputs R, G and B of the RGB color space, the PNs that fired in (1), were classified as Fired Pixel Neurons (FPN). These FPNs connected themselves to their neighboring FPNs using the 8-connected neighborhood technique, diagrammatically represented in Fig. 3. In effect, the connection mechanism accomplishes an output similar to the one shown in Fig. 4.

In this region-clustering mechanism, effective edge-detection algorithms help to separate nearby mutually exclusive clusters. The connected FPNs indicate the road region to be the one that is closest to a baseline. A baseline is a fixed horizontal line below which the image of the camera shows information of the vehicle it is attached to (See Fig. 11 later). This baseline, in images that do not have such information, is the last row of the image. From the selected road region, the respective region cluster is the final neural structure of the RSB from which the INSAV obtains the road structure, depicted in Fig. 5. This structure provides other information, like the bend of the road or an approaching cross-road, etc.

i-1,j-1	**i-1,j**	**i-1,j+1**
i,j-1	**i,j**	**i,j+1**
i+1,j-1	**i+1,j**	**i+1,j+1**

Fig. 3: The yellow cells are the FPNs which get connected to FPN at (i,j).

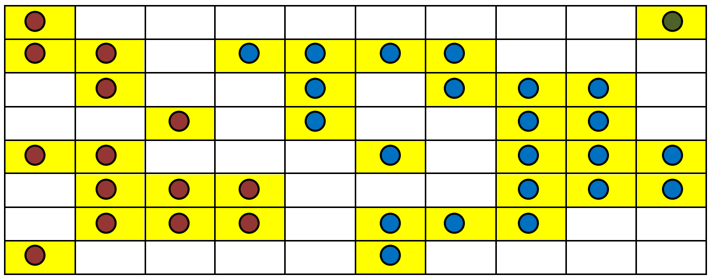

Fig. 4: Desired FPN clustering algorithm output.

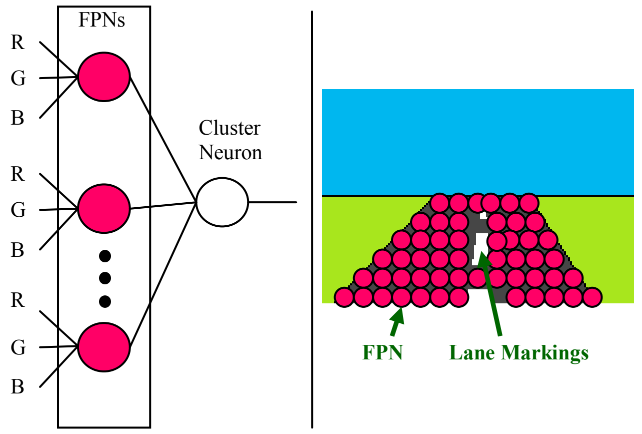

Fig. 5: Fired Pixel Neuron (FPN) based Neural Road Structure.

The road structure obtained is used to learn road colors in later stages or environments. Hence the algorithm considers only the cases when the vehicle is in close vicinity to a road. Using such cases, the new image frame, $\boldsymbol{I_{new}}$, is sampled at the known road structure. A new road sphere, $\boldsymbol{S_{new}}$, is then trained using the obtained pixel colors in $\boldsymbol{I_{new}}$, based on the initially provided sphere, ***S***. First a new COG, COG$_{Snew}$, is found. This COG is calculated as the pixel color closest to the COG of ***S***, COG$_S$, by Euclidean Distance (ED). If the road structure is defined as a set, $\boldsymbol{R_{struct}}$, of pixel locations that define the structure, then

$$COG_{Snew} = min\left[ED\left(I_{new(i,j)}(R,G,B), COG_S\right)\right] \quad \forall\,(i,j) \in R_{struct} \tag{2}$$

The new radius is obtained by initially normalizing all pixel colors in $\boldsymbol{R_{struct}}$. This was done to identify like-colors to the COG$_{Snew}$. The normalization is based on the maximum color channel R, G or B as this provides a specific intensity independent color signature, unlike the method used in [24].

$$\begin{aligned} R_{norm(i,j)} &= R_{(i,j)}/max\left(R_{(i,j)}, G_{(i,j)}, B_{(i,j)}\right) \\ G_{norm(i,j)} &= G_{(i,j)}/max\left(R_{(i,j)}, G_{(i,j)}, B_{(i,j)}\right) \\ B_{norm(i,j)} &= B_{(i,j)}/max\left(R_{(i,j)}, G_{(i,j)}, B_{(i,j)}\right) \end{aligned} \tag{3}$$

The EDs of these normalized colors are then calculated with the normalized COG$_{Snew}$. A threshold T=0.2, set based on ~90% of collected ED data, selects all normalized pixel colors below it for minimum intensity evaluation. This is based on the assumption that the road color is of a lower intensity. The ED between the winning pixel's color and the COG$_{Snew}$ is taken to be the radius. This new road sphere, $\boldsymbol{S_{new}}$, is somewhere close to the initial road sphere, ***S***, on the RGB space. The $\boldsymbol{S_{new}}$ is then used for identifying the road in $\boldsymbol{I_{new}}$. Equations (1) and (2) are evaluated again. The pseudo code for this algorithm is provided in Fig. 6.

```
Pseudo Code:

 1) Initialize FPN frame $F[sizeof(I)] = 0$;
 2) Using $\boldsymbol{S}$, $if\ I_{(i,j)}(R,G,B) \in S,\ F(i,j) = 1$;
                        $else\ F(i,j) = 0$;
 3) Cluster pixels in $F$ using 8-connected neighborhood.
 4) The set $\boldsymbol{R_{struct}}$ = The cluster nearest to the baseline.
 5) Sample $\boldsymbol{I_{new}}$ at $\boldsymbol{R_{struct}}$.
 6) Compute $COG_{Snew}$ using (2).
 7) Normalize pixel colors sampled in 5) using (3).
 8) Compute vector D = ED of normalized colors in 7)
    with the normalized $COG_{Snew}$.
 9) Create a new vector N, s.t $if\ D(i) \le T, D(i) \in N$
                              $\forall\ i \in [1, |D|]$
10) Store corresponding intensity values of vector N, in 9),
    in a vector T. (The intensity value is the denominator
    of (3).)
11) Denormalize the color in N with the least intensity.
12) Compute the radius of $\boldsymbol{S_{new}}$ as the ED between the
    color in 11) and the $COG_{Snew}$.
13) Repeat 2) on $\boldsymbol{I_{new}}$ with the new sphere $\boldsymbol{S_{new}}$.
14) Repeat 3) and 4).
```

Fig. 6: Pseudo Code for Road Structuring Algorithm.

B. *Neural Path Tracking*

Immediately after determining the road structure, Path Tracking is done. Assuming the road ahead is straight and not curved, the Path Tracking Algorithm (PTA) calculates the Vanishing Point (VP) on the image using an unsupervised approach, hence no training set. A supervised approach like the Least-Mean-Square Algorithm cannot be used because there is no separation line required but an alignment on the road boundary points from the RSB. As mentioned earlier, the VP has uses in motion classification of objects in the scene. Typical Path Tracking algorithms use Mobile Robot Dynamics [15]-[20]. These typical systems, however, work with laser sensors to obtain the environment.

The PTA initiates by selecting the all the Road Boundary Points (RBP) from the determined road structure from the RSB. These RBPs then form lines with other RBPs in parallel. These lines have slopes, m_i, and y-intercepts, c_i, where the i represents the RBP index. Since we are assuming the road is straight ahead and not curved, there exists a partition between left-side RBPs and right side RBPs. Effectively the left-side RBPs connect to only the left-side ones, while the right-side RBPs vice versa. Two neurons, i.e. the Left Road Boundary Neuron and Right Road Boundary Neuron, learn alignments of the road boundaries on the left-side and right side. The point of intersection is the VP, as depicted in Fig. 7.

Although the road is straight, in many cases the RBPs are not in a straight line form. This is because other vehicles on the road obscure the road boundaries. In order to tackle such cases, the PTA initially computes the weighted average form of the slope vectors with the slope samples obtained previously. These Weighted Average Slope Vectors (WASV) eliminate unlikely slopes by bringing them closer to the more likely slopes. Fig. 8 shows the effect of WASV on a slope

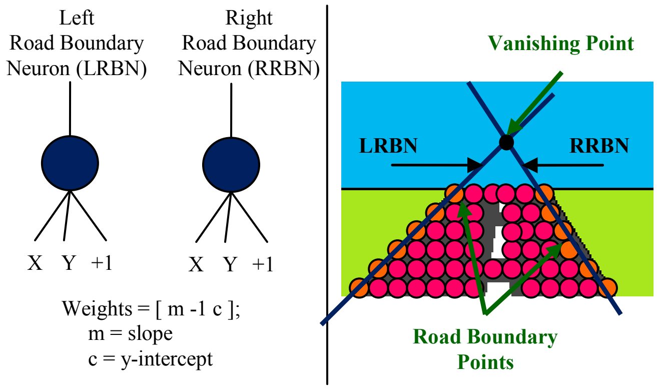

Fig. 7: Path Tracking Neurons.

vector, S. Because the road boundaries are expected to be disfigured, this computation becomes very useful in the faster prediction of the Actual Road Boundary Slope (ARBS). It is in fact used as an initialization step for the ARBS. The weights used in the WASV computation are based on absolute differences (AD), or a single dimension ED, between two vector scalars. Their range lies between 0 and 1. Usually one would quantify this to be an exponential value with negative power of the AD, as in Radial Basis Functions. However, in order to simplify this for hardware suitability, the weights are quantified to be some function of the reciprocal of AD. This function avoids the weights from reaching infinity and keeps them between 0 and 1.

A simple model of two parallel resistors, with impedance $Z = \frac{R_1 R_2}{R_1 + R_2}$, was used in enclosing the range of the weights. The impedance Z always lies below the $min(R_1, R_2)$. Using this model, the smallest resistor for the weight function is equal to 1Ω. So if

$$r = |val_1 - val_2| \tag{4}$$

then,

$$W(val_1 - val_2) = w = 1/(r+1) \tag{5}$$

where w is the assigned weight. The WASV is then computed as follows. The initiated mean, μ_0, is taken as the first value in the slope vector, S with samples s_i, of interest. Then,

$$\mu_t = \mu_{i-1} + 0.5 \times (s_i - \mu_{i-1})$$
$$w_i = W(\mu_t - \mu_{i-1})$$
$$\mu_i = \mu_{i-1} + w_i \times 0.5 \times (\mu_t - \mu_{i-1})$$

$$\Longrightarrow \mu_i = \mu_{i-1} + w_i \times 0.25 \times (s_i - \mu_{i-1}) \tag{6}$$

Referring back to Fig. 8, note that the strong variations on the slope values are eliminated. After the WASV computation, the slope of the road boundary is predicted. The initial predicted slope is the average value of the WASV. The updated prediction of the slope is based on the following concept. Let the WASV be a vector V.

$$\widetilde{w}_i = \frac{\sum_j W_j(\widehat{m}_{i-1} - v_j)}{|V|} \approx W(m_{act} - \widehat{m}_{i-1}) \tag{7}$$

Here

$\widetilde{w}_i$ = approx. AD weight of predicted slope from the actual slope.

$\widehat{m}_{i-1}$= predicted slope at iteration i-1.

$|V|$ = cardinality of vector V.

v_j = j^{th} element of V.

m_{act}= the actual slope of the road boundary.

Following from (8), the updated predicted slope at iteration i, was calculated as follows.

$$\widehat{m}_i = \widehat{m}_{i-1} \pm \left(1 - \frac{1}{\widetilde{w}_i}\right) \tag{8}$$

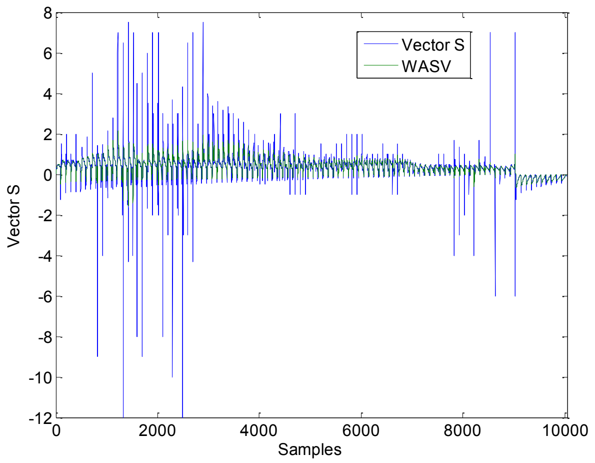

Fig. 8: Effect of WASV on vector S.

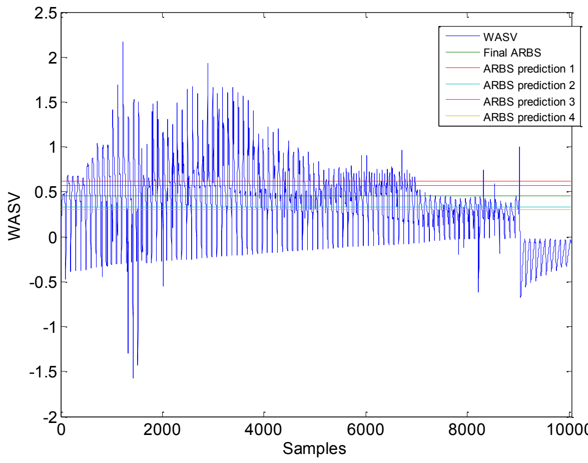

Fig. 9: Prediction of ARBS.

Equation (8) says that two values of slopes were predicted. The one with the maximum $\widetilde{w}$, calculated using (7), stayed as the predicted one for next iteration. Due to the nature of the WASV, the predicted values of the slopes exhibited oscillations between iterations. Fortunately they were oscillating among a fixed mean value which was the almost equivalent to the ARBS. Therefore, predicted slope values for four consecutive iterations were taken and averaged. This averaged value was reported as the final prediction of ARBS. Fig. 9 shows the prediction mechanism of the ARBS.

Its corresponding y-intercept was similarly extracted using (7) and (8). But instead of a weighted average vector initialization using (6), the y-intercept scalars, from the y-intercept vector, were accentuated based on the predicted ARBS. In other words, the closer the scalars in the slope vector S were to the predicted ARBS, the more their corresponding y-intercepts were accentuated. To enhance the accentuation, the square of the AD values were taken for weight computation. Due to this enhancement, a really strong threshold, of about 0.9999, was capable of extracting only the significant y-intercept scalars. Fig. 10 provides the pseudo code for this algorithm.

```
Pseudo Code:

1)  Select road boundary points.
2)  Form lines that hold slope and y-intercept scalars.
3)  Calculate the WASV using (6).
4)  Calculate average of WASV as the initial predicted
    ARBS.
5)  Calculate $\widetilde{w}_i$ from eqn. (7).
6)  Update predicted ARBS from (8) and (7).
7)  Average four predicted ARBS to be final ARBS.
8)  Compute accentuated weight vector C for y-intercept
    vector using squared AD.
9)  Create new vector Y, s.t $if\ c_i > 0.9999, c_i \in Y;$
10) Repeat 5) – 7) similarly for Y.
```

Fig. 10: Pseudo Code for Path Tracking Algorithm.

IV. Experimental Results

The following section will demonstrate the obtained results from the implementation of the algorithms explained in the previous section.

A. Neural Road Detection and Training (Road Structuring)

The road detection for the provided road color sphere, ***S***, was robust. The trained $\boldsymbol{S_{new}}$, too, provided high percentage of accuracy in road detection. The percentage of accuracy was measured based on summing Type I and Type II Errors of road detection in the Region-Of-Interest (ROI). By definition, Type I Errors are false positives when the Null Hypothesis, H_0, is rejected and Type II Errors are false negatives when H_0 rejection fails. In the road detection accuracy measurement the H_0 is,

$$\mathcal{H}_0: Non\ Road\ Region\ in\ ROI$$
$$\mathcal{H}_1: Road\ Region\ in\ ROI$$

Then the total error of detection, e, is,

$$e = \frac{Type\ I\ Error + Type\ II\ Error}{\mathcal{H}_0 + \mathcal{H}_1}$$

$$\%\ Accuracy = (1 - e) \times 100 \tag{9}$$

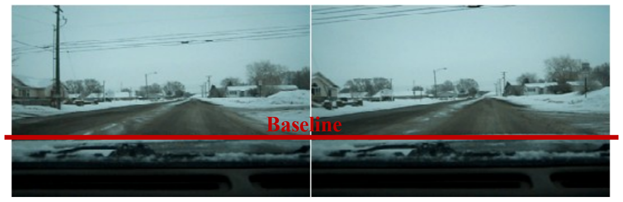

Fig. 11: Training video sequence frames 1(left) and 30(right).

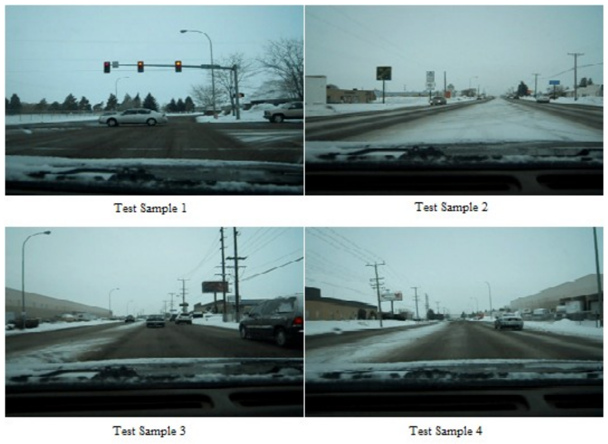

Fig. 12. The four test sets.

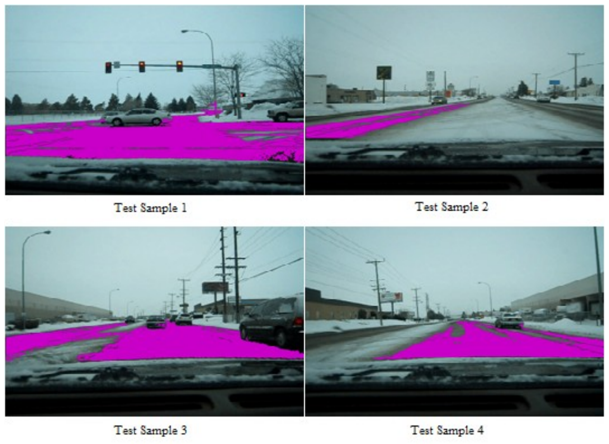

Fig. 13. Test Results on the four test sets.

Results demonstrated an average of 93 percent accuracy in identifying the road on four test sets of 30 video sequence frames from one training set of 30 video sequence frames. The sample training video sequence is shown in Fig. 11 and the four sample test video frames are shown in Fig. 12. The sample results of the algorithm on the four test sets are shown in Fig. 13. The success of this algorithm proved useful for the PTA. The algorithm proved its independence of varying road structures of later environments. This makes it less prone to false negatives or false positives.

The tested images depict a single type of environment, i.e. snow roads under moderate illumination. During the driving period, this type of situation of having similar environments occurs most of the time. Therefore the algorithm guarantees acceptable performance in such situations.

B. *Path Tracking Algorithm*

Due to the success of the RSB, the PTA too proved robustness. The accuracy measure used for this algorithm was based on the simple relative error calculations of pixel positions. Since the algorithm is an unsupervised type, there are no training sets.

$$Error = \frac{\Delta x}{x} + \frac{\Delta y}{y}$$

$$\%\,Accuracy = (1 - Error) \times 100 \qquad (10)$$

This algorithm proved an average of 91 percent accuracy of vanishing point estimation with high speed of prediction. The success of the road structuring algorithm was the key to its high performance.

Fig. 14 shows four tested straight roads that ran independent of each other. The blue lines, the neurons, in the figure show their alignments on the road boundaries. The point of intersection of the lines, marked as a black spot, is the estimated vanishing point. The yellow spots in the images of the third row show the actual vanishing point of the images because of the significant error in the estimated vanishing point.

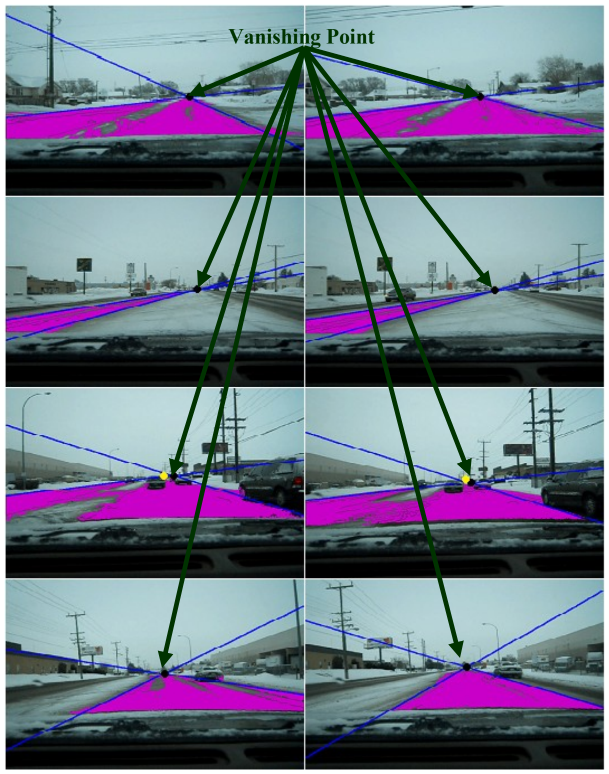

Fig. 14. Results of the PTA Algorithm

V. Conclusion

The most important blocks of the generic INSAV, i.e. the Road Structuring and Path Tracking Blocks, were successfully designed. These blocks also feature simplified computations suitable for hardware implementation. The results of the current work, presented in this paper, are encouraging towards the realization of the whole INSAV for autonomous vehicle hardware.

The next step is to design the Motion Classifier and Obstacle Detection Blocks. A further challenge lies in making the blocks more interactive with every other block. This type of interaction ensures much more robustness to completely varying environments.

VI. References

[1] E.D. Dickmanns, "The development of machine vision for road vehicles in the last decade," in *Proc. IEEE Intelligent Vehicles'02 Conference*, 2002.

[2] D. Erez , M. Ofer, P.S. Gideon and S. Amnon, "Forward Collision Warning with a single camera," in *Intelligent Vehicles Symposium*, 2004, pp. 37–42.

[3] L. Frederic, V. Dizan, F. Thierry and L. Christian, "Avoiding Cars and Pedestrians using Velocity Obstacles and Motion Prediction," in *Intelligent Vehicles Symposium*, 2004, pp. 375–379.

[4] C. Wohler and J.K. Anlauf, "Real-time object recognition on image sequences with the adaptable time delay neural network algorithm – applications for autonomous vehicles," in *Image and Vision Computing*, vol. 19, 2001, pp. 513–618.

[5] M. Foedisch and A. Takeuchi, "Adaptive Real-time Road Detection Using Neural Networks," in *Proc.7th International IEEE Conf. on Intelligent Transportation Systems*, 2004, pp. 167–172.

[6] C-T Tudoran and V-E Neagoe, "A New Neural Network Approach for Visual Autonomous Road Following," in *Latest Trends on Computers*, vol. 1, 2010, pp. 266–271.

[7] B. Leibe, K. Schindler, N. Cornelis and L.V. Gool, "Coupled Object Detection and Tracking from Static Cameras and Moving Vehicles," in *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, 2008, pp. 1683–1689.

[8] D. Feiden and R. Tetzlaff, "Obstacle Detection in Planar Worlds using Cellular Neural Networks," in *Proc.7th IEEE International Workshop on Cellular Neural Network Applications*, 2002, pp. 383–390.

[9] S. Cherng, C-Y Fang, C-P Chen and S-W Chen, "Critical Motion Detection of Nearby Vehicles in a Vision-Based Driver-Assistance System," in *IEEE Trans. Intelligent Transportation Systems*, vol. 10, 2009, pp. 70–82.

[10] H-C Choi and S-Y Oh, "Illumination Invariant Lane Color Recognition by using Road Color Reference & Neural Networks," in *Proc. Intl. Joint Conf. on Neural Networks*, 2010, pp. 1–5.

[11] K. Ito, N. Ichihara, H Inoue, R. Fujita and M. Yasushi; Pioneer Corp., "Car Navigation System with Image Recognition," in *Digest of Technical Papers Intl. Conf. on Consumer Electronics*, 2009, pp. 1–2.

[12] W. Yanqing, C. Deyun, S. Chaoxia and W. Peidong, "Vision-Based Road Detection by Monte Carlo Method," in *Information Technology Journal, 9:*, 2010, pp. 481–487.

[13] M. Bertozzi and A. Broggi, "GOLD: A Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection," in *IEEE Trans. Image Processing*, vol. 7, no. 1, 1998, pp. 62–81.

[14] Z. Li, L. Li and Y. Zhang, "IVS 09: Future Research in Vehicle Vision Systems," in *IEEE Intelligent Systems Magazine*, vol. 24, issue 6, 2009, pp. 62–65.

[15] O. Mohareri, R. Dhaouadi and M.M. Shirazi, "Intelligent neural network based controllers for path tracking of wheeled mobile robots: A comparative analysis," in *IEEE International Workshop on Robotics and Sensor Environments*, 2010, pp. 1–6.

[16] J. Gao, J Zhu, B Wei and S. Wang, "Unmanned Vehicles Intelligent Control Methods Research," in *Proc. 9th ICEMI*, 2009, pp. 3-736–3-741.

[17] R-J Wai and C-M Liu, "Design of Dynamic Petri Recurrent-Fuzzy-Neural-Network for Robust Path Tracking Control of Mobile Robot," in *Proc. Intl. Joint Conf. on Neural Networks*, 2010, pp. 1–8.

[18] V.S.K. Chaitanya and P.K. Sarkar, "A Neural Network Algorithm for the Error Optimization in the Path Tracking Control of a Mobile Robot," in *Proc. Intl. Joint Conf. on Neural Networks*, 2006, pp. 2501–2507.

[19] M. Harb, R. Abielmona, E. Petriu and K. Naji, "Neural Control System of a Mobile Robot," in *Proc. Intl. Joint Conf. on Neural Networks*, 2008, pp. 2825–2832.

[20] X. Xu, H. Zhang, B. Dai and H-G He, "Self-learning Path-tracking Control of Autonomous Vehicles Using Kernel-based Approximate Dynamic Programming," in *Proc. Intl. Joint Conf. on Neural Networks*, 2008, pp. 2182–2189.

[21] J. Canny, "A Computational Approach to Edge Detection," in *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no.6, 1986, pp. 679–698.

[22] R.L. Gregory, *Eye and Brain*, 3rd ed. New York: McGraw-Hill, 1978.

[23] R.S. Snell, *Clinical Neuroanatomy for Medical Students*, 3rd ed. Boston, MA: Little, Brown, 1992.

[24] I.K. Konstantin and E.D. Dickmanns, "A Color Vision System for Real-Time Analysis of Road Scenes," in *Intelligent Vehicles Symposium*, 2004, pp. 54–59.