Chapter 3

Omnidirectional Vision for Road Understanding*

* Zhigang Zhu, Shiqiang Yang, Guangyou Xu, Xueyin Lin, Dingji Shi, **Fast Road Classification and Orientation Estimation Using Omni-View Images and Neural Networks,** *IEEE Transactions on Image Processing*, vol 7, no 8, August 1998: pp 1182-1197

Abstract

This paper presents the results of integrating omnidirectional view image analysis and a set of adaptive backpropagation networks to understand the outdoor road scene by a mobile robot. Both the road orientations used for robot heading and the road categories used for robot localization are determined by the integrated system, the RUNN (Road Understanding Neural Networks). Classification is performed before orientation estimation so that the system can deal with road images with different types effectively and efficiently. An omni-view image sensor captures images with 360 degree view around the robot in real-time. The rotation-invariant image features are extracted by a series of image transformations, and serve as the inputs of a road classification network (RCN). Each road category has its own road orientation network (RON), and the classification result (the road category) activates the corresponding RON to estimate the road orientation of the input image. Several design issues, including the network model, the selection of input data, the number of the hidden units and the learning problems are studied. The internal representations of the networks are carefully analyzed. Experimental results with real scene images show that the method is fast and robust.

I. INTRODUCTION

An autonomous mobile robot (vehicle) should have three basic functions in order to move safely in an outdoor road environment: road following, obstacle avoidance and landmark recognition. All of them need the comprehensive understanding of the natural road scene. In this paper we deal with the first and part of the last issues in an integrated manner. A robot moves along the road and makes decisions when it reaches some pre-defined points. It should obtain two kinds of information from the visual sensors: the robot heading (or the road orientation) and its location (in terms of road categories, e.g., straight road, intersection or T-junction). So the tasks can be regarded as road classification and orientation identification.

Two problems prevent the existing vision algorithms and systems from being successfully used in this real world application. Firstly, most previous vision systems of mobile robots can only view objects in front of them due to the narrow view angle of the commonly used TV cameras. As a result, robots may go astray or collide against objects from the side or behind. Secondly, most of the vision algorithms for outdoor road understanding only work well in predefined environments. However, whenever the environment changes, they may perform improperly.

To solve the first problem, several researchers have studied the omnidirectional vision system. Elkins and Hall [1] used fish eye lens for the visual navigation of an outdoor mobile robot. The mobile robot located itself by referring to the known targets in the environment. Yagi et al. [2] applied a conic mirror to acquire omnidirectional image for the indoor mobile robot, and vertical edges were used to detect obstacles while the robot carried out a constant linear motion on flat floor. Hong et al [3] used a spherical mirror to capture the omnidirectional images and studied the image-based homing problem in the indoor environment. The 1D horizon circle of the omnidirectional image captured at a location was matched with those of a series of predefined "homing" locations and guided the robot to reach the nearest "home". Stein and Medioni [4] used a rotating camera to acquire the 360-degree panoramic images in the terrain. The omni-directional curves of the horizon were used to find "drop off" location of the robot. Most of the above approaches share the same characteristics that specific features in the omnidirectional images are used to solve the given problems in predefined environments. Omnidirectional image methods applied to the outdoor road scenes need further investigation.

Artificial neural networks (ANN) are a reasonable solution for the second problem due to the following two reasons. First, for the real world problem of road understanding, there are various aspects that should be taken into consideration, such as the weather, the light, static and dynamic objects on the road, noise, and so on. Therefore it is difficult for a vision algorithm designed by a human programmer to include all kinds of varieties. Moreover the need of giving thresholds for feature extraction and parameter estimation often bothers the researchers and engineers in the image analysis of real world problems. Neural networks, in contrast with being programmed, capture knowledge and skills by training. Second, there are enough data to train the ANNs. While the robot moves along the road, image sequences are captured at the rate of 25 (or 30) frames per second. For certain road segment with similar surroundings (e.g., straight paved road with trees and bushes on both sides), plenty of representative sample images with similar characteristics can be obtained.

ANN architectures for early vision, especially motion perception, have been proposed based on physiological as well as psychological evidences [5,6]. In the well known ALVINN (Autonomous Land Vehicle In a Neural Net) project, a full-sized self-driving van equipped with video cameras and four onboard workstations has been developed and built at CMU [7], which employ the ANN in real world application. ALVINN is a fully connected three-layered backpropagation network , whose input is 32x32 sub-smapled road image from a video camera and whose output is the vehicle heading (1 out of 45) required to make the van stay on the road. Nine hidden units are used in the system and the output is updated 15 times per second. The ALVINN network is trained using a unique “on the fly” procedure. Road image is processed as the vehicle is driven by a person down a highway. Vehicle headings, while steered by the human driver, provide the feedback necessary for training. Although the ALVINN has successfully driven the Navlab vehicle on various types of the road in various weather conditions, the system is still not perfect. The images taken at different viewing directions by a commonly used TV camera are quite different from each other due to the perspective projection and quite different viewing zones, so the network will be complicated for complex scene. In their recent work, Jochem and Pomerleau [8,9] proposed the so-called virtual camera method to handle the lane transition problem of highway driving. The basic idea is to find the suitable image sub-region and to transform it as if it was captured by a virtual camera from the desired viewing point. As a result the input requirements of the ALVINN are satisfied. Problem may also arise when the vehicle head for directions that are not included in the training range, or when the road scenes vary drastically during the long driving.

Recently we have proposed a method in which omnidirectional imagery and the neural networks are combined in order to reach a better solution for the aforementioned problems[10,11]. The omnidirectional images provide rotation-invariant features to the neural networks, while the neural networks provide an adaptive way for image classification and identification. Our work, sharing the similar goals with the CMU’s ALVINN, possess four distinctive features:

(1) The system estimates not only the headings of the robot but also (at the first stage) the road types along the route.

(2) Omni-directional view image is used. In this way the system will not be troubled by the limited view angle of the camera that has brought lot of problems in the past works of road following.

(3) Rotation-invariant features are extracted from rotation-dependence omni-directional images by a series of image transformations. No image segmentation and explicit feature extraction are needed.

(4) Since we use the pre-processed image data as the inputs of the network, the number of the input variables is reduced and the networks are concise. The separation of the process of the road classification and the heading decision also greatly reduces the computation complexity.

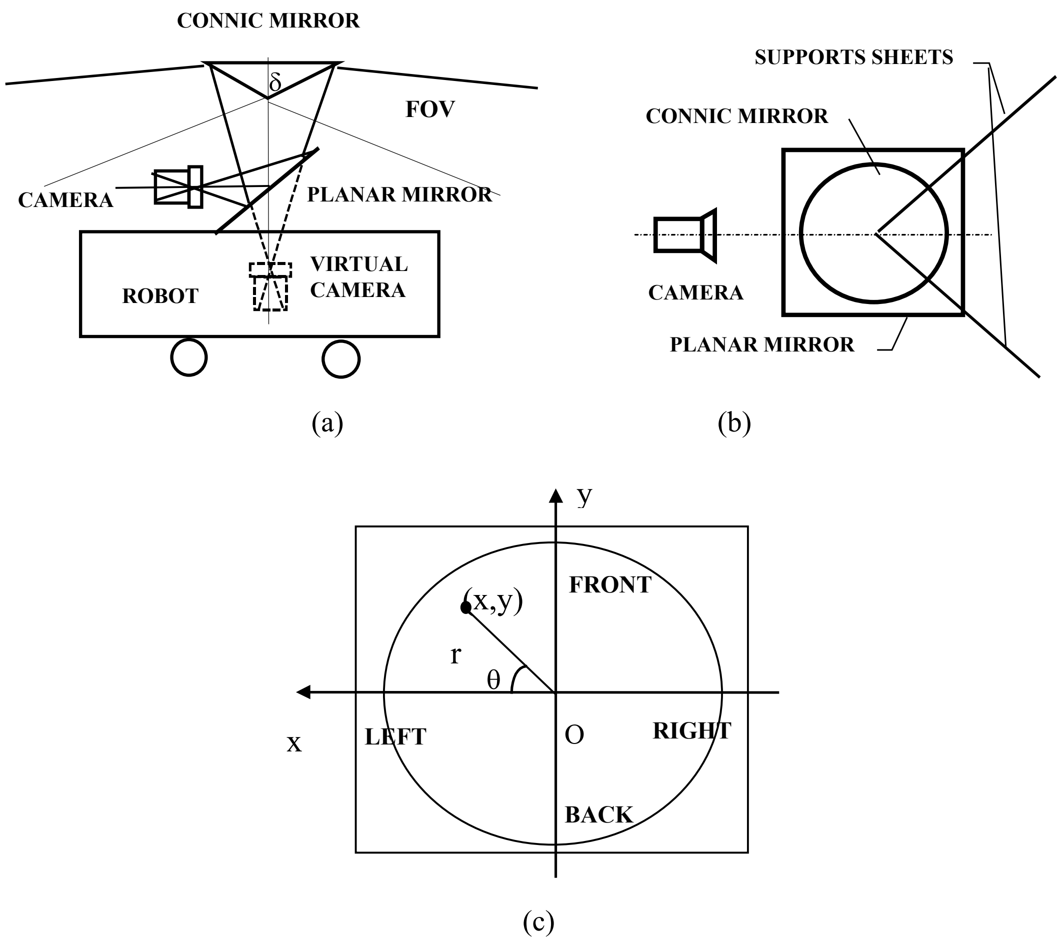
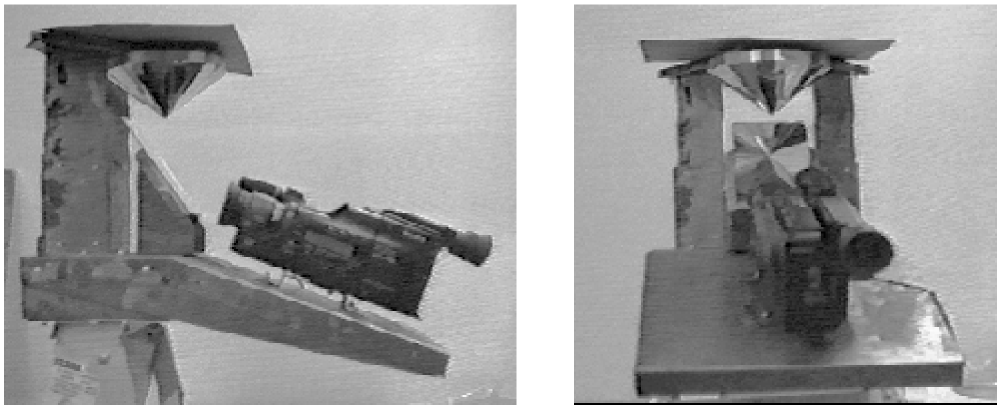


Fig. 1. Omni-view image sensor. (a) Side view (supports not shown). (b) Top view (supports shown) (c) Image coordinates

II. OMNI-VIEW IMAGING SENSOR

To capture the omnidirectional view (omni-view) image of the environment, various imaging methods have been explored, including fish-eye lens [1], conic mirror [2], spherical mirror [3] and rotating camera [4]. The time-consuming omni-view imaging by using a rotating camera prevents it from applying to real-time problem. In order to obtain in real-time the omni-view of the road and objects around the mobile robot, the rest of the methods may be used. A fish eye lens yields a wide semi-spherical view around the camera. However the image of roads and objects near the robot locates along the circular image boundary with poor image resolution. Imagery taken by a spherical mirror provide a similar omnidirectional view of the environment as by a conic mirror, but a large part of the image is occupied by the robot itself and the image along the radius axis is not purely perspective projection but includes a quadratic distortion. So we adopt and modify the conic projection sensor system COPIS proposed by Yagi et al [2], aiming to deal with the situation of the outdoor road scene.

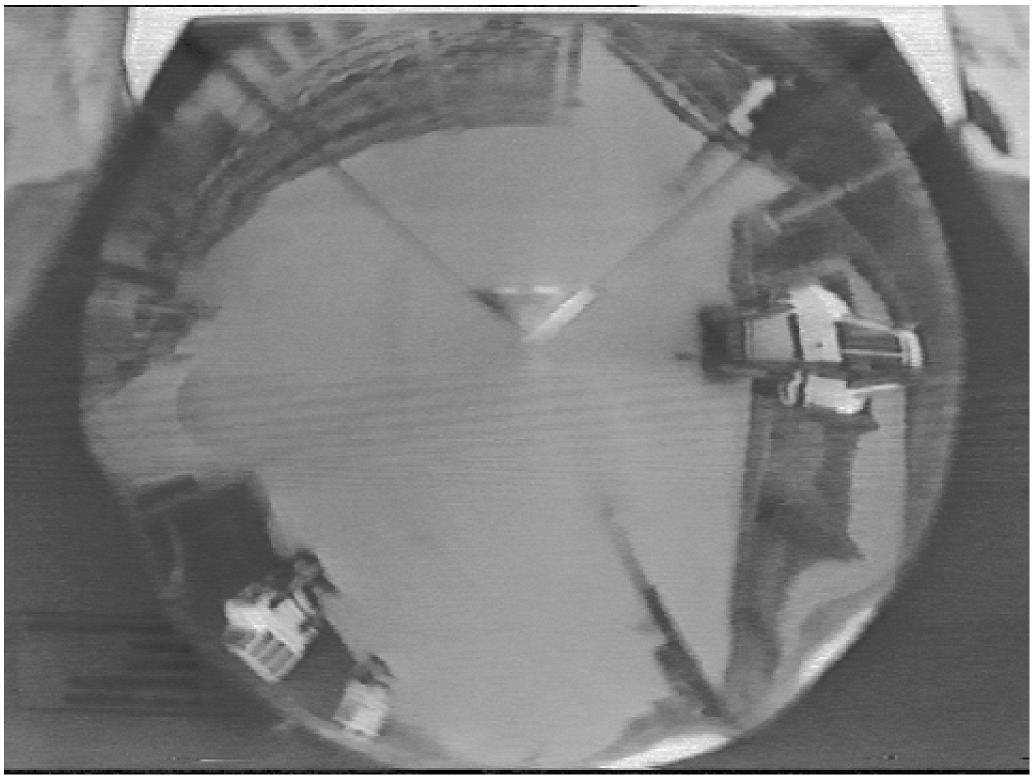
The geometry and configurations of the omni-view image (OVI) sensor are shown in Fig. 1. A conic mirror (with a vertex angle δ=55°) is fixed on the roof of the robot by two very thin sheet mental supports whose thickness is 1 mm. The intersection line of the sheets coincides with the vertical axis of the conic mirror. A planar mirror is placed beneath the conic mirror with a tilt angle $\pi/4$. The camera is mounted horizontally on the roof of the robot. The optical axis of the camera, the vertical axis of the conic mirror, and the normal of the planar mirror lie in the same vertical plane. The distance between the conic mirror and the roof of the robot must be large enough to avoid the occlusion of the field of view by the robot. The position of the planar mirror and the camera should be carefully adjusted to ensure the coincidence of the optical axis of the "virtual camera" inside the planar mirror and the vertical axis of the conic mirror. We use a planar mirror and a horizontally placed camera instead of a camera pointing upward for sake of easy installation and adjustments, and the protection of the camera lens. Moreover, the omni-view image after two mirror reflections gives un-inverse view of the scene, as opposite to the mirror image by the COPIS system. The transparent tube used in the COPIS is replaced by two thin sheet mental supports of the conic mirror because we have found that the commonly available transparent tube is not completely transparent, and the reflection by the tube troubles image analysis, which is more severe in the outdoor environment. The diameter of the conic mirror is 110 mm and the nearest edge of each thin sheet is 100 mm far away from the conic vertical axis, so each sheet occupies less than 0.6 degrees out of the 360-degree's field of view. Fig. 2 shows a prototype of the omni-view image (OVI) sensor used in our experiments. The camera is placed on a titled plane (about 15 degrees) instead of a horizontal plane in order to avoid the occlusion of the field of view by the large-size camcorder used in our experiment. The tilt angle of the planar mirror increases to about 60 degrees correspondingly.



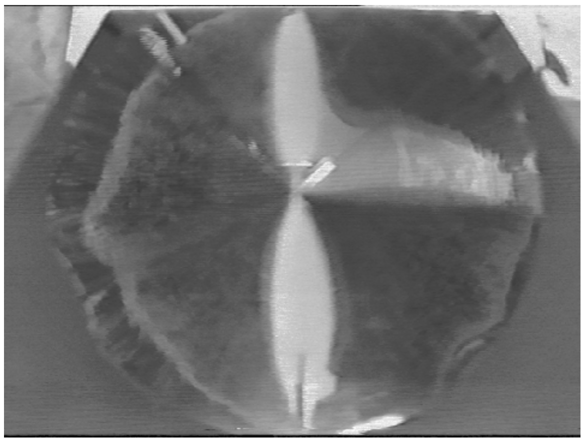
(a) (b)

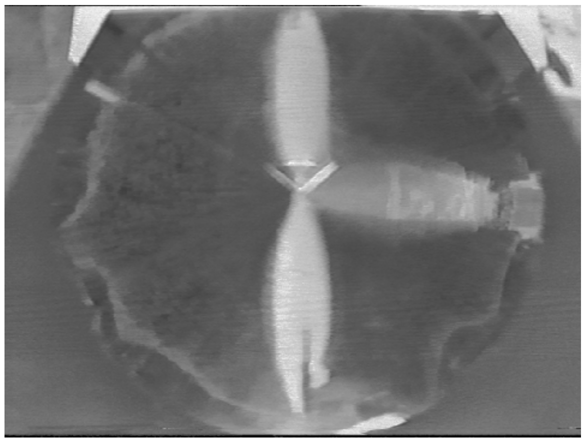
Fig 2. A prototype of the omni-view image (OVI) sensor. (a) Side view (b) Back view.

The image taken by our OVI sensor represents a 360° view of the scene around the robot, ranging from about 5 meters to 30 meters on the ground. Omni-directional image taken by a conic mirror is equivalent to the image taken by a tilted line scan camera rotating around a vertical axis, while it optical center is moving along a circle around the same axis [12]. Fig. 3 shows three omni-view images of different scenes: one is captured in the center of a square, the second is near the entry of a T-junction, and the last is in the center of the T respectively. As the prototype OVI sensor is not accurately adjusted and the conic mirror is not perfect, the sheet supports project two thin gray lines (see Fig 3a) and there are some geometric distortions around the boundaries of the images. Fig. 4 shows the corresponding ground projections of the OVI images in Fig. 3. The ground projection is approximately equivalent to the image taken by a down-looking camera high above the robot [12]. The parallelism of the road has not been completely recovered due to the errors stemmed from coarse system adjustments and a rough calibration procedure.



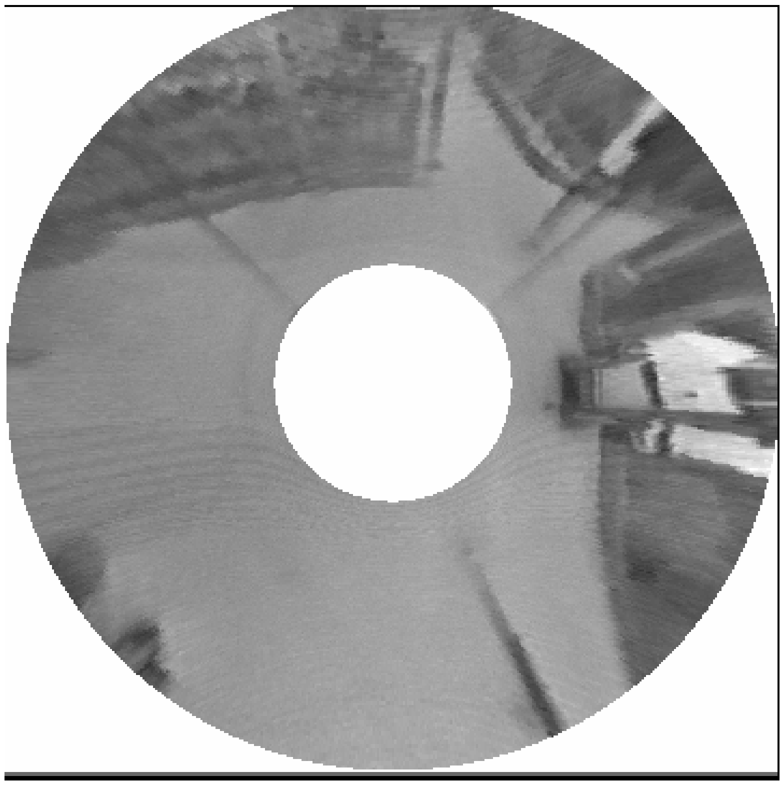
(a)



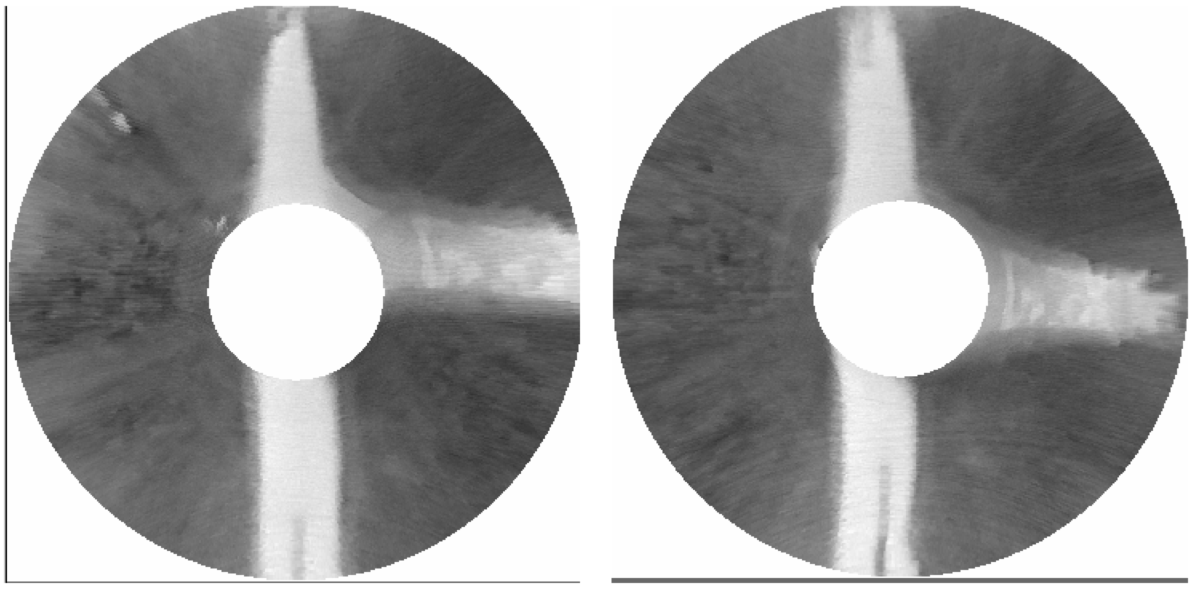


(b) (c)

Fig.3. Omni-view images. (a) 512x512 OVI image of a square scene (lower-left is a car , right is a truck and upper-right is a person). (b),(c) 256x256 images captured when the robot just entered the T-junction and was in center of the T.



(a)



(b) (c)

Fig.4. Ground projections of images in Fig. 3. The robot is in the center of the white disc in each image.

Although the resolution of the OVIs is relatively low compared with images of a commonly used TV camera, the 360° view image has some distinctive advantages when it is used in the road scene understanding: (1) It covers all the information in the scene around the robot. As a result, the robot never misses the road. (2) The image is of rotation invariance in the sense that the structure of the image and the field of view are not changed at all if the robot rotates around the optical axis of the camera, no matter what kinds of 3D structures of the environment are. (3) The low-resolution sensing image is quite fit for the qualitative recognition (classification) of road categories. A small amount of lateral offsets of the robot on the road with moderate width, for example, do not bring great changes in the omni-view image. Appearances in the image remain similar if the robot moves within the same road segment (same category) surrounded by similar scenes. As an example, images with different rotation and lateral offsets for a paved straight road are shown in Fig. 5.

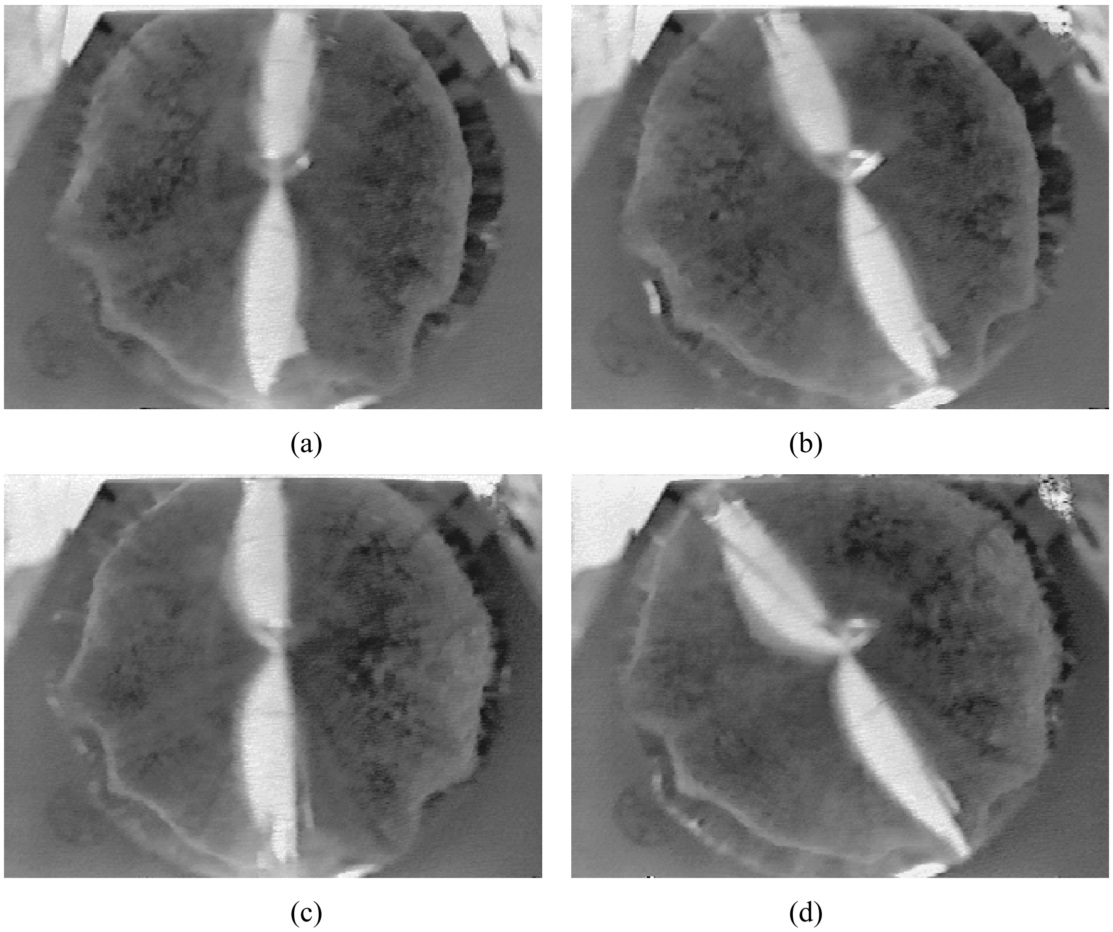


Fig. 5. Images of a paved straight road. The robot (a) headed forward, (b) rotated for an angle, (c) moved to the roadside and then (d) rotated again. The width of the road is about 5 meters.

III. ROTATION-INVARIANT FEATURES

A. Polar Transform and Projection Transform

Suppose the origin of the OVI coordinate system xoy is in the center of the image where the conic vertex is projected, we transform the Cartesian coordinate image I(x,y) into a polar coordinate image (r,θ) (Fig. 1c):

$$r = \sqrt{x^2 + y^2}\ ,\ \theta = \tan^{-1}(y/x) \qquad (1)$$

where r is the radius and θ is the orientation angle(0 - 2π). Ideally, for a 256x 256 original sensor image, the resolution of the angle in the polar image is about 1 degree, so the dimension of corresponding polar image is 128×360 (r=0,…,127; θ=0,...,359). Since the center of the OVI is not in the image center in our actual system, and the near-center zone is too blur to be useful, the effective range of radius is from 30 to 100. Fig. 6 shows the polar images of the OVIs shown in Fig.5.

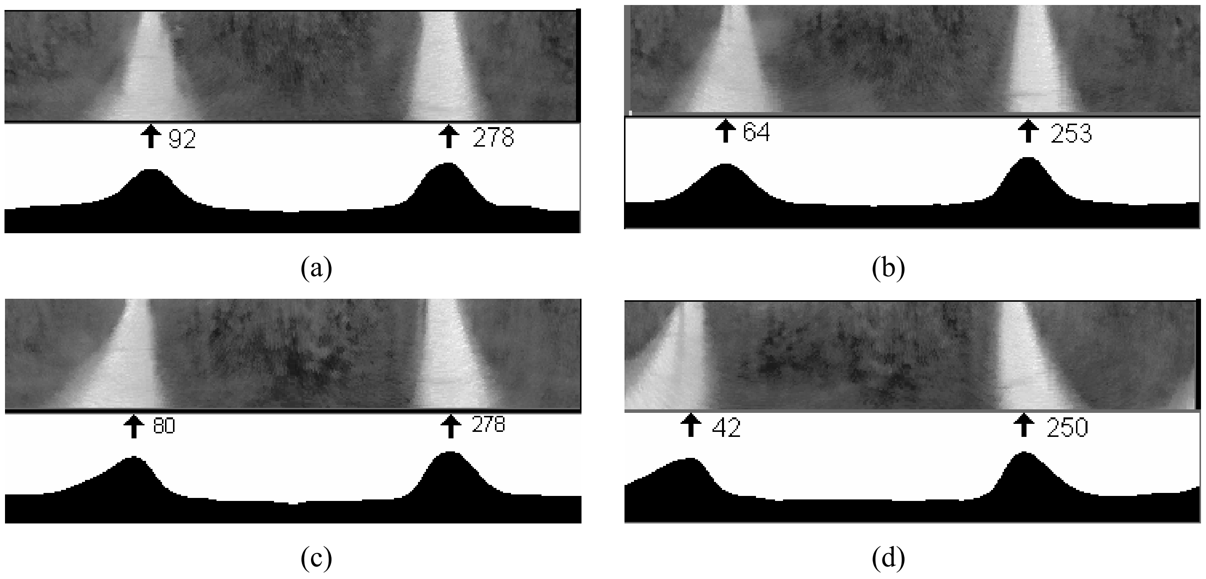


Fig. 6. Omni-view image transformations. (a), (b), (c) and (d) show the polar transform images(upper row) and the smoothed orientational projections(lower row) for the corresponding images in Fig. 5 respectively. The orientational projections have been smoothed for the sake of peak finding. The arrows between the images of the upper and lower rows indicate the estimated center of the roads, while the number beside them are the corresponding orientation angles in degrees.

The 2D polar image I(r, θ) is transformed to a 1D orientational projection u(θ) by accumulating (projecting) the image along radius r (Fig. 6) as

$$u(\theta) = \sum_{r} I(r,\theta),\ \theta \in [0,2\pi) \qquad (2)$$

In this paper we use the Fourier transform of the original projection to extract the rotation-independent features, instead of determining the road orientation in the preprocessing stage as we did in [10], which may bring errors to the samples. The orientation of the road will be estimated after the road category has been correctly classified.

B. Road Orientation for Data Collecting

In order to prepare the samples for training and testing the BP networks, we should know the ground truth of the road categories and orientations of each image at first. The former can be easily provided by the human operators since the image sequence for a certain road category lasts for a long time period. The later, however, should be estimated by the computer automatically due to that road orientations change from frame to frame (see Section V). We use a simple three-step peak finding and tracking algorithm[12] to determine the road orientations if the roads show intuitive peaks in the orientational projection u(θ). The polar image, orientational projection and the estimated road orientation for each omni-view image in Fig. 5 are shown in Fig. 6. Estimated angles between the roads in front of and behind the robot are 186, 189, 198 and 208 degrees for the four images respectively. Even if the peak finding and tracking method is tolerant of the geometric distortions of the images, it should be pointed out that the method partially relies on the peak finding procedure. For some road categories, the method may be not successful (refer to Fig. 12). So other techniques need to be investigated.

C. Rotation Invariant and Dependent Features

Suppose that the orientational projection u(θ) is sampled to N discrete orientations, and then normalized into **U** = {u(n), n=0,...,N-1} with zero mean and unity standard deviation. The normalized procedure eliminates or at least reduces the influence of any illumination changes of images that are captured at different times. If rotation angle of the robot is

$$\phi = -\frac{2\pi n_0}{N} \qquad (3)$$

where $\phi \in [-2\pi,0]$, then the new orientational projection v(n) will be the circular shift of u(n) by n_0, and can be denoted as

$$v(n) = u(n - n_0) \qquad (4)$$

where u(n) is the orientational projection when the robot heads for the front road (refer to Fig. 5 and Fig. 6). The Fourier transform of u(n) is

$$a(k) = \frac{1}{N}\sum_{n=0}^{N-1} u(n)\exp(-\frac{j2\pi kn}{N}),\ \mathrm{k = 0,...,N-1} \quad (5)$$

and the Fourier transform of v(n) can be expressed as

$$b(k) = a(k)\exp(-\frac{j2\pi n_0 k}{N}), k = 0,1,..., N-1 \quad (6)$$

By representing a(k) and b(k) in amplitude-phase forms

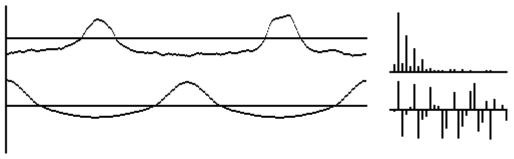
$$\begin{matrix} a(k) = a_k \exp(j\psi_k) \\ b(k) = b_k \exp(j\varphi_k) \end{matrix},\ \mathrm{k=0,...,N-1} \quad (7)$$

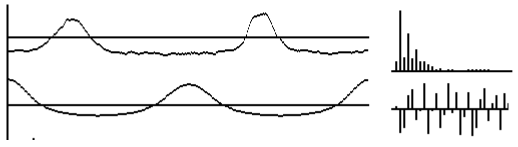
we have the following results:

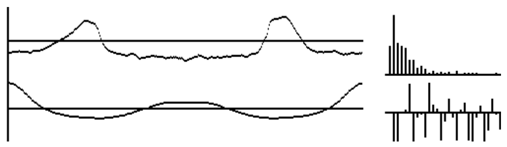
$$b_k = a_k, k = 0,..., N-1 \quad (8)$$

$$e^{j\psi_k} = e^{j(\varphi_k + k\phi)} \quad (9)$$

$$\psi_k = 2\pi m_k + \varphi_k + k\phi\ ,\ \mathrm{k=1,...,N-1} \quad (10)$$


(a)


(b)


(c)

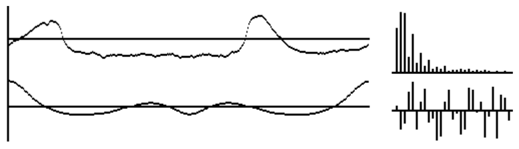

(d)

Fig. 7. Each of (a), (b), (c) and (d) shows the normalized projection (N=360, upper left), the first 30 Fourier amplitudes (upper right), the first 30 Fourier phases (lower right) and the auto-correlation functions ACF(lower left)) of the corresponding polar images in Fig. 6. The ACFs are rotation-invariant, which is clearly shown in (a) and (b). Since there are large geometric distortions between projection in (d) and the others, the ACFs are not identical.

where m_k is an integer which indicates $2\pi m_k$ additive ambiguous in the kth phase value. Equation (8) says that the Fourier amplitudes are invariant to the rotation of the omni-view images. Therefore they are appropriate features for road scene classifications. Equation (9) and (10) give the basic relation to estimate the orientation difference between two omni-view images. For real scene images, the equality can not validate exactly. Analysis and experimental results show that the Fourier phases are very sensitive to the noises, especially to the geometrical distortion of the orientational projections. Fig. 7 shows the normalized projection (N=360), the first 30 Fourier amplitudes and phases of the polar images in Fig. 6. It can be seen that the Fourier amplitudes are similar for the four images, but the phases are not stable, especially when the corresponding amplitudes are small.

D. Road Orientation Difference for Data Collecting

Actually, the orientation difference could be estimated by searching the minimum value of the following distance

$$d(\phi) = \sum_{k=0}^{N-1} (a_k e^{j\psi_k} - b_k e^{j(\varphi_k + k\phi)})^2 \qquad (11)$$

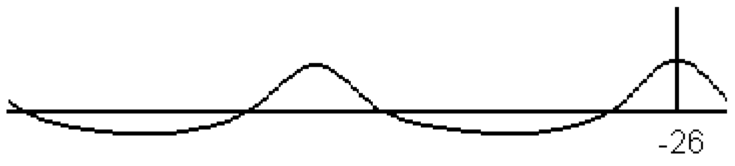
for each $\phi(n_0) = \phi = -\frac{2\pi n_0}{N}$, $n_0=0,1,\ldots,N-1$. The computing complex of this procedure is $O(N^2)$. This is equivalent to find the maximum value of the circular cross-correlation function (CCF)

$$C(n_0) = \sum_{n=0}^{N-1} u(n) v((n+n_0) \text{ modulo } N),\ n_0 = 0,1,\ldots,N-1 \quad (12)$$

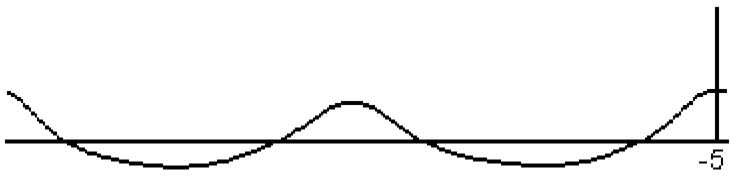
which is also $O(N^2)$ computation. Since we have obtained the Fourier transform of u(n) and v(n), we hope to use them for the estimation of orientation difference. The correlation theorem states that the (circular) correlation of two real signal sequence u(n) and v(n) is equal to the inverse discrete Fourier transform (DFT) of the product of conjugation of DFT a(k) and DFT b(k), i.e.,

$$C(n_0) = F^{-1}(a^*(k) b(k)) \ (13)$$

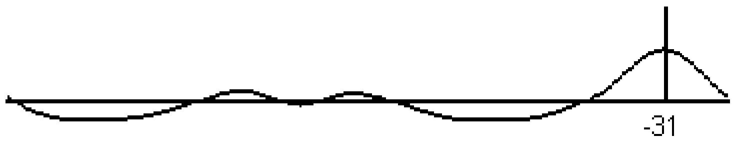
The alternative approach has only $O(3N\log_2 N)$-time complexity. The advantage of the global correlation method is that no feature extraction is needed. So correlation method is more robust to noise and more general in different cases. Fig. 8 shows the experimental results. Compared with the estimations in Fig.6, the results from CCF method are the average of the orientation difference between the front roads and the orientation difference between the back roads in the two images. Therefore the CCF method, together with the peak finding and tracking method, is used in the data collection and training for the networks since the behavior of the robot can be controlled in the training stage, and the man-machine interaction can be involved. In the real operations, the orientation is estimated using the neural networks.



(a)



(b)



(c)

Fig. 8. Orientation differences by using cross-correlation function (CCF) in the Fourier domain. (a) CCF between projections (a) and (b) in Fig. 7 . (b) CCF between projections (a) and (c) in Fig. 7. (c) CCF between projections (c) and (d) in Fig. 7. The vertical lines and the number besides them indicate the position(n_0) where maximum CCF take place (The axis is 0 to -360 from right to left).

E. Another Rotation Invariant Feature

From equations (12) and (13) we can obtained another rotation-invariant sequence, the auto-correlation function (ACF) of u(n)

$$R(n_0) = \sum_{n=0}^{N-1} u(n)v((n+n_0)\,\text{modulo}\,N),\ \mathrm{n}_0 = 0,1,\ldots,N-1 \quad (14)$$

The ACF of u(n) is equal to the inverse Fourier transform of the energy spectrum of u(n)

$$R(n_0) = F^{-1}(a^*(k)a(k)) = F^{-1}(a_k^2) \qquad (15)$$

Fig. 7 also shows the ACFs estimated by the inverse Fourier transforms of the energy spectrums of the projections. Compared with the Fourier amplitudes, the similarity of ACFs is more intuitive (refer also to Fig. 13). So the ACF can serve as another rotation invariant input for road classification.

IV. SYSTEM ARCHITECTURE

In order successfully work with real-world problems, we must deal with some design issues, including the network model, network size, activation function, learning parameters, and selection of training samples. We will address these issues in this and the following sections, bearing in mind that we face the problems in outdoor road scene, namely the road classification and orientation estimation.

A. RUNN Architecture

It is commonly accepted that the backpropagation learning procedure has become the most popular method to train the multi-layer feed-forward networks [13], and the so called backpropagation (BP) networks have been widely used in character recognition, speech recognition, vehicular control and many more cases of applications[7,10,13,14]. There are two main schemes for using ANNs in a pattern classification system[15]. The first one employs an explicit feature extractor (not necessarily a neural network). The extracted features are passed to the input stage of the multi-layer BP network. The scheme is very flexible in incorporating a large variety of features. However explicit feature, e.g. boundary of the road, has proved to be very difficult to extract in the outdoor road scene. The other scheme does not explicitly extract features from the raw data. The feature extraction implicitly takes place within the hidden layers of the ANN. A nice property of this scheme is that feature extraction and classification are integrated and trained simultaneously to produce optimal classification results. However it is not clear whether the types of features that can be extracted by this integrated architecture are the most effective for the given pattern classification problem. Moreover, this scheme requires a much larger network than the first one. A typical example of the second scheme for visual navigation is the ALVINN[7-9].

We take an alternative approach from the two typical schemes. The basic model for Road Understanding Neural Networks(RUNN) is an adaptive combination of an image processing module (IPM) and several fully connected BP networks --- a single three-layer Road Classification Network (RCN), a two-layer Road Orientation Network (RON) for each road category. Fig. 9a shows the system architecture. Raw image data is preprocessed by the IPM before feeding into the neural nets. However no image segmentation and explicit feature extraction are needed. A composed macro-network, composed of several basic BP networks, is constructed to solve both the road classification and orientation estimation.

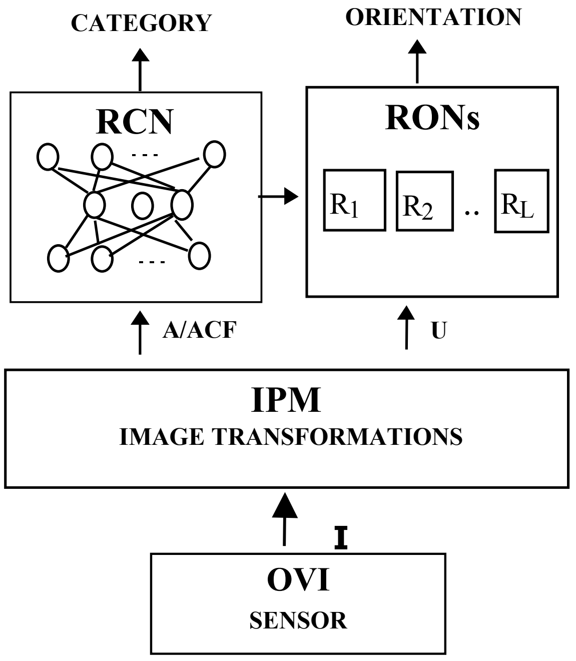
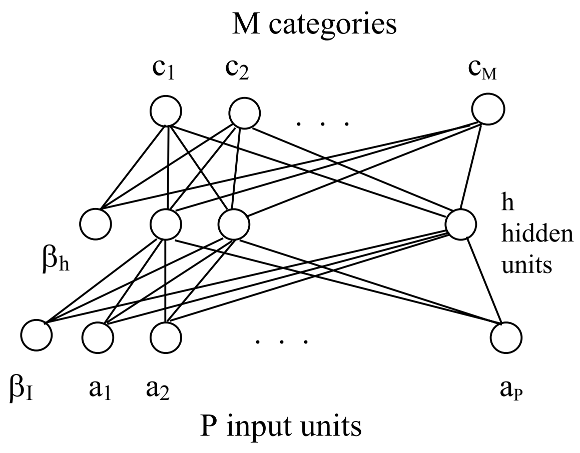
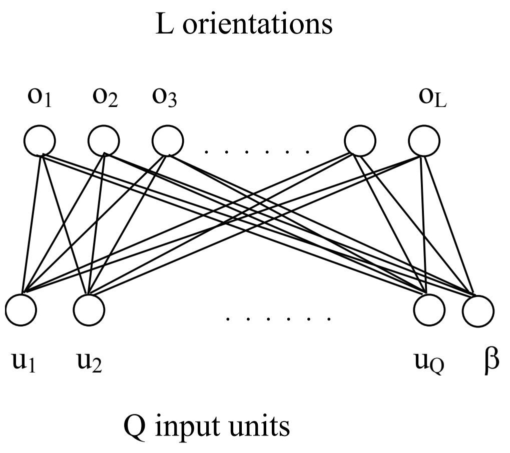


Fig. 9a. The Architecture of RUNN. The system is composed of an Omni-View Image (OVI) sensor, an Image Processing Module(IPM) , a Road Classification Network (RCN) and a set of Road Orientation Networks (RONs), R_1,R_2, …, R_L.





Fi.9b. RCN for road recognition   Fig. 9c. RON for road orientation

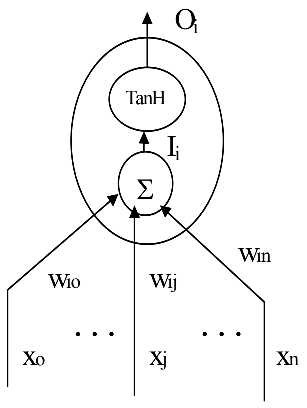


Fig. 9d. The model of PE

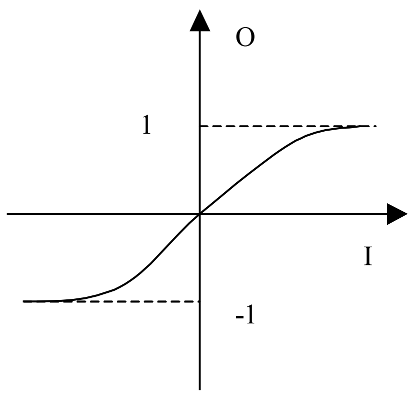


Fig. 9e. Transfer function TanH

B. Configurations of RCN and RONs

We adopt the convention that a standard Y-layer backpropagation (BP) network consists of an input layer, (Y-2) hidden layers and an output layer of units successively connected in a feed-forward fashion. The RCN is a fully connected three-layer BP network (Fig. 9b). The inputs of the RCN are P ($\leq$N) rotation-invariant image data (i.e. Fourier amplitudes or ACF), and the outputs are M road categories. The net can be viewed as a nonlinear input-output mapping. The connection between the input and the hidden layers extract special features of input patterns and the connections between the hidden and the output layers recognize specific road categories. Therefore the hidden units may represent different kinds of features and the number of units in this layer will be decided by an experiment-based approach. A bias unit is connected to the hidden layer and the output layer respectively.

Each RON is a two-layer BP network without any hidden layer (Fig. 9c). Its function is virtually a correlation operation, and details of this design will be explained in Section VI. The inputs of each RON are Q ($\leq$N) rotation-dependence image data (i.e. the original orientational projection), and the outputs are L road orientations.

The basic model of the neuron unit, or the processing element (PE) for all the networks, is shown in Fig. 9d. The summation function is

$$I_i = \sum_j w_{ij} x_j + \beta \qquad (16)$$

where i stands for the current PE, j stands for a PE that i is connected to, x_j is the output of PE j, w_{ij} is the weight of the connection of i and j, and β is the bias value. The transfer function of each input unit is linear and that of each hidden or output unit is the hyperbolic tangent (TanH, see Fig. 9e):

$$I' = I * gain, O = \frac{e^{I'} - e^{-I'}}{e^{I'} + e^{-I'}} \quad (17)$$

where gain is called the steepness factor. The TanH is quite similar to the Sigmoid transfer function. However its range is -1 to +1, as opposed to the Sigmoid range of 0 to 1. Because the output of the transfer function is used as a multiplier in the weight update equations, a range of 0 to 1 could lead to a bias to learning higher desired output (approaching 1). The TanH gives equal weight to low- and high- end values.

C. Installation and Execution

The RCN and RONs are set up using *Nworks* tool[15], which provides a variety of ANN architectures and the flexibility of parameter controls. The IPM's hardware is a pipeline image processing machine named PIPE [16], hosted by a PC 486 with a Nworks environment. The UserIO interface (written in C) of the Nworks connects the image processing module (PIPE) and the neural network environment, working in parallel. Our PIPE system includes a video stage, an input stage, three modular processing stages(MPSs), an iconic-to-symbolic mapping stage (ISMAP) and a output stage. Each of the MPSs includes several frame buffers, two real time convolution processors, ALUs, and most importantly, a two-valued function(TVF) LUT that greatly facilitated the image geometrical transformations. The polar transformation can be carried out in a MPS at the video field rate (60 fields per second). The projection transformation is implemented in the ISMAP at the frame rate. The 1D Fourier transform is carried out by the PC486.

The RUNN works in the following three steps.

1. The omni-view image is captured and transformed by the IPM. The orientational projection **U** and rotation-invariant Fourier amplitudes $\mathbf{A} = \{a_k, k = 0,..., N-1\}$ are obtained. For comparison, the auto-correlation function (ACF) sequence $R = \{r_k, k = 0,..., N-1\}$ is also calculated using the inverse Fourier transform of the energy spectrum.

2. The rotation-invariant data (**A** or ACF) is used to decide the road category by the RCN.

3. The road category estimation is used to activate the correct RON, and the rotation-dependent data (i.e. the original orientational projection **U**) is fed into that selected RON to estimate the road orientation.

There are two advantages of the separation of the road classification and orientation estimations. First, since rotation invariant data are used as the input of the RCN instead of the original image, the distinctiveness of the input units is increased, and therefore the complexity of the network is reduced. If the Fourier amplitude A is used as the input of the RCN, the number of the input units can be reduced to $P \le \frac{N}{2}$. Second, since a separate RON is used to estimate the road orientation for each road category, and the classification result is used to select the corresponding RON, the efficiency of the networks will be improved.

V. DATA COLLECTION FOR TRAINING AND TESTING

The data for the RUNN are collected while the robot is moving on the road. In our experiments, the robot moves along the route around the Main Building at the campus of Tsinghua University. The omni-view image sequences are recorded by a video camcorder, and then are played back for processing by the RUNN system.

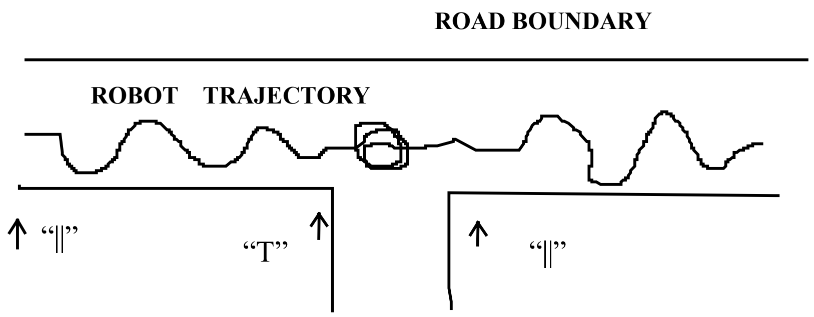
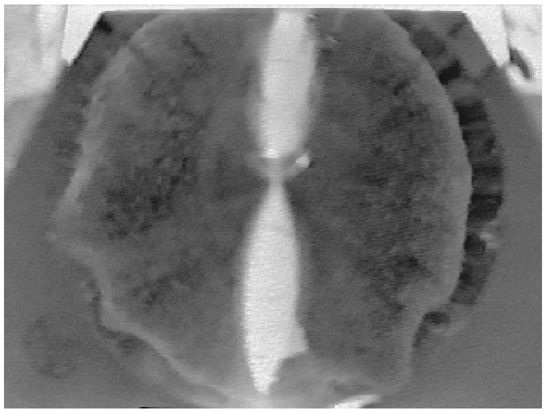
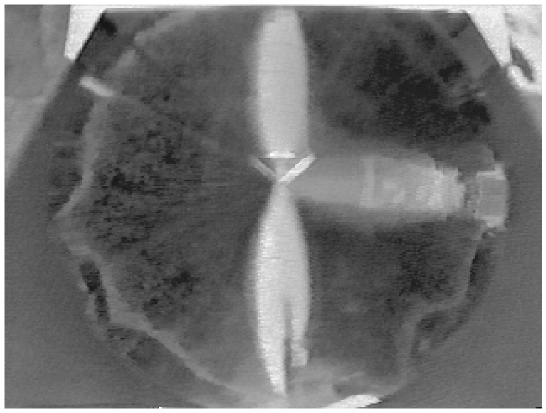


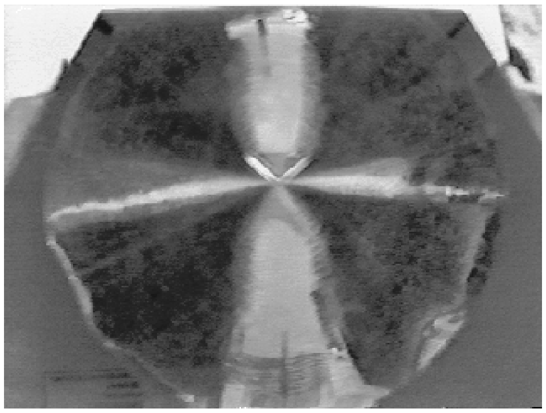
Fig. 10. Collecting the data for training and testing



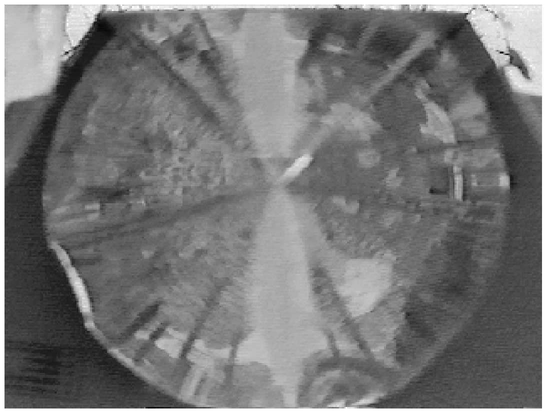
(a)



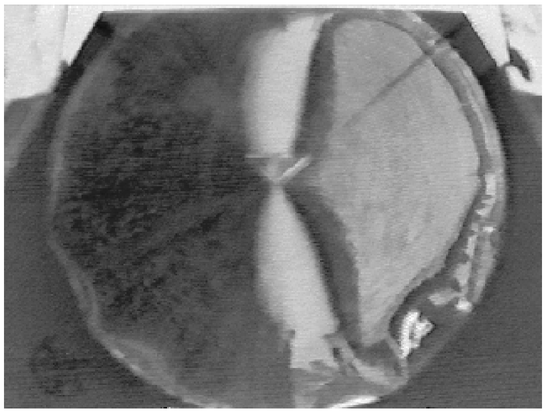
(b)



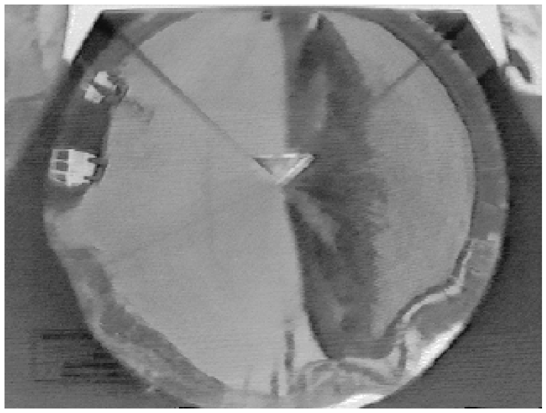
(c)



(d)



(e)



(f)

Fig 11. Sample images of the six road categories (a) Paved straight road (“||”). (b) T-junction (“T”). (c) Intersection (“+”). (d) Earthy road(“D”). (e) Curved road (“C”). (f) Square (“S”)

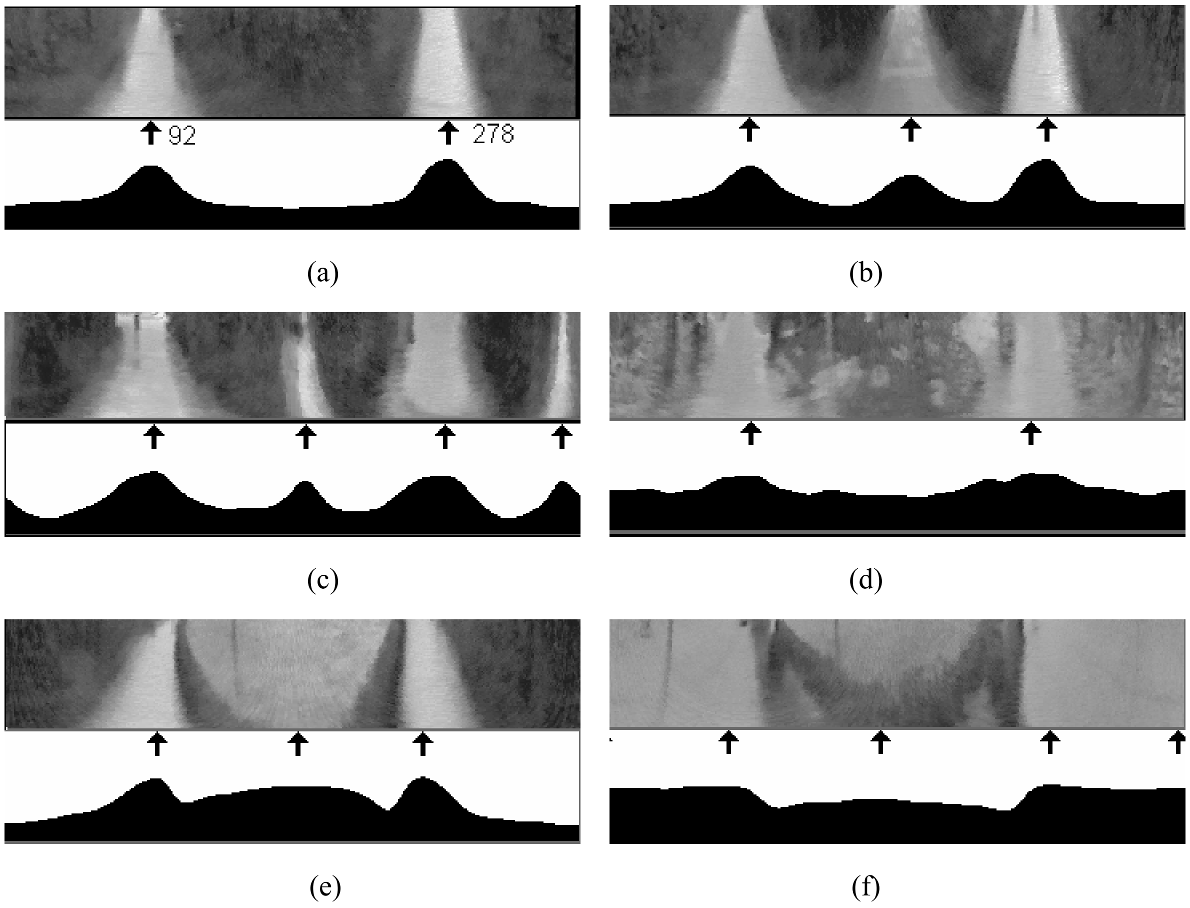


Fig.12. Polar images and projections. Each of (a), (b), (c), (d), (e) and (f) shows the polar transform image(upper row) and the smoothed orientational projection(lower row) for the corresponding image in Fig. 12 . The arrows between the upper and lower images indicate the center of the roads, while the number beside them are the corresponding orientation angles in degrees. Notice that the some of the results are not correct for images in (e) and (f).

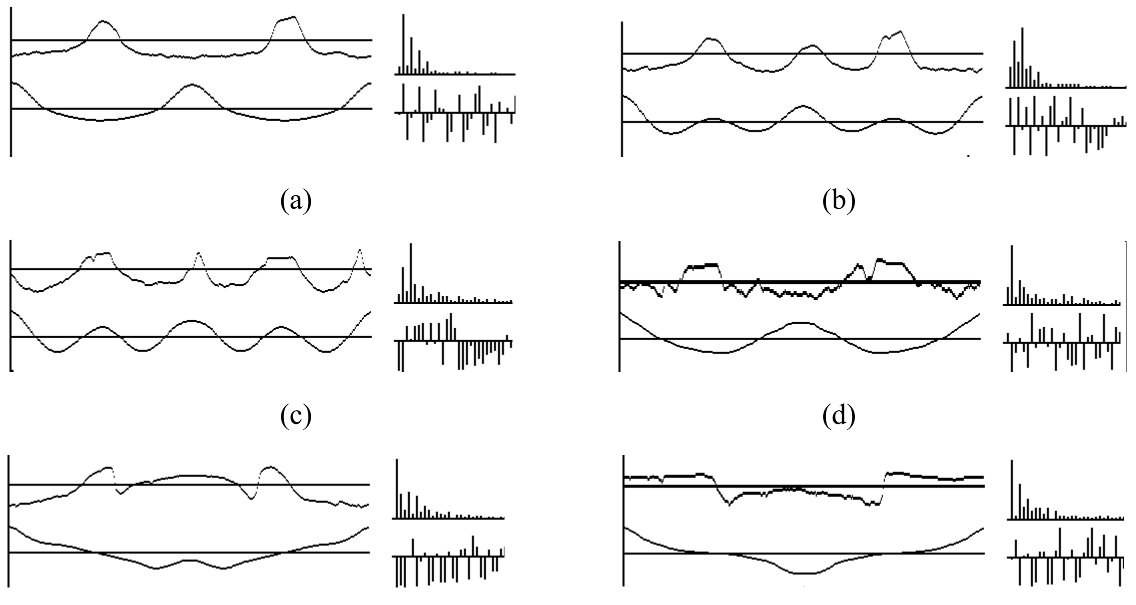

(a) (b) (c) (d) (e) (f)

Fig. 13. The normalized projection (N=360, upper left), the first 30 Fourier amplitudes (upper right) the first 30 Fourier phases (lower right) and the ACF (lower left) of the corresponding polar image of each category in Fig. 12. Notice the differences of Fourier amplitudes and ACF curves among the six categories, and the similarities among the sampled images of the same category in Fig. 7.

A. Collecting the Data for the RUNN

At the beginning of data collecting for each category of road segment in a camera shot, the robot heads for the front road (i.e., the road orientation angle is 0), and the desired outputs of the RCN, representing the road categories, are assigned by human supervisor. The peak finding and tracking algorithm (Section III) with human interaction verifies the orientation of the first frame for each shot. For the current experiments, the input data of the RCN are N=32 sub-sampled elements of the orientational projection. The road images are classified as 6 categories (M=6): paved straight road surrounded by bushes and trees (denoted as “||”), T road junction(denoted as “T”), intersection(denoted as “+”), earthy road surrounded by grass and trees(denoted as “D”), narrow curved road passing through the garden in front of the building(denoted as “C”) and the square in front the Main Building(denoted as “S”). In order to cover most of the situations, the robot zigzags on the road so that captured images can cover most possible directions and various lateral offsets (refer to Fig. 10). For preparing data to train and test the RON, the orientation difference is calculated for the successive image frames within the same road category by find the maximum value of the CCF in equation (13). The absolute road orientation is obtained by accumulating the orientation differences and modified by the peak-finding and tracking method. In order to cover most of the rotation (orientation) cases, the sampled orientational projection is shifted by software to simulate all the different road orientations. Both the inputs to the network as well as the desired outputs are mapped into numbers. Fig. 11 shows one typical sample image for each of the six categories. Fig. 12 and Fig.13 show the corresponding polar images, projections, Fourier amplitudes and phases, and the auto-correlation functions of the images in Fig. 11.

Table 1. Selecting and dividing the data for the RCN

\ Category Set \	||	T	+	D	C	S	Total
Original set	1073	371	182	557	441	175	2799
Selected set	930	312	160	445	374	148	2342
Training set	*133*	*133*	*133*	*134*	*133*	*133*	*799*
Testing set	*797*	*179*	*26*	*311*	*241*	*15*	*1569*

B. Selecting and Dividing the Data

It is important to make sure that examples selected for training the network do not have any dubious data fields (e.g., outliers). To this end, we calculate the mean Fourier amplitude (MA) vector of all the samples within one category, and the distance between any Fourier amplitude vector of each sample and the mean MA is used to judge whether it is a "good" example. Good examples are chosen from the original raw data set and then are divided into the training set and the testing set. For best results, the selection of training set is based on the following rules: (1) Every category has roughly same amount of examples. (2) The training set is reasonably representative of each category. (3) It is best to make the testing and training sets completely separate. The actual selection and division results are listed in Table 1. The numbers listed in the table are the numbers of real images captured by the OVI sensor, and for the training and testing of the RCN. For the training and testing of the RONs, each sample has as many as L rotated versions.

C. Scaling the data

The inputs are already in number forms. The desired outputs are set to either 0 or 1 (e.g., the output vector is "1, 0, 0, 0, 0, 0" for category "||"). Since we use the TanH as transfer function, we will need to scale these values between -1 to +1. Fortunately a so-called MinMax Table mechanism is provided by the Nworks tool. This preprocessing facility computes the lows and highs of each data field corresponding to each input unit (in the training set or both the training and testing sets) and stores in the MinMax Table. The Nworks then computes the proper scale and offset for each data field. Real world values are then scaled to network range(-1 to +1) for presenting to the network . Whenever a scaled result is produced, it is de-scaled to real world units.

VI. CONSTRUCTION OF THE RUNN

We construct the RUNN during the training and testing process using real image data, and study the following four issues: (1) the suitable representation of input data, (2) the number of the hidden units, (3) the internal representation of the networks; and (4) the learning problem, for example, the epoch size, the converging speed, etc. .

A. Learning Rule and Schedules

The back-propagation learning strategy is used to training the network with M output units. The error in the output layer is computed as the difference between the desired output ($\mathbf{D}=(d_1,d_2,...,d_M)$) and the actual output ($\mathbf{Y}=(y_1,y_2,...,y_M)$). This error E, transformed(scaled) by the derivative of the transfer function TanH, is back-propagated to the priori layer where it is accumulated. This back-propagated and transformed error becomes the error term of that priori layer. The process of the backpropagation continues until the first layer is reached.

In our implementation, the normalized cumulative delta learning rule[16] is used for the RUNN. Cumulative generalize delta rule attempts to alleviate the problem of structured presentation of the training set. The basic idea is to accumulate the weight changes over several training presentations and make the application all at once. The update equations are

$$m'_{ij} = m_{ij} + C_1 e_i x_{ij}, \text{ at each iteration;}$$
$$\begin{cases} w'_{ij} = w_{ij} + m_{ij} + C_2 a_{ij} \\ a'_{ij} = m_{ij}, & \text{, if LCNT mod AUX1} = 0 \\ m'_{ij} = 0 \end{cases} \quad (18)$$

where C_1 is the learning coefficient (step size), C_2 is the momentum factor, $\mathbf{E} = (e_1,...,e_n)$ is the error vector (as described above), $\mathbf{X_i} = (x_{i0},...,x_{in})$ is the inputs to the ith PE in the current layer, $\mathbf{W}_i = (w_{i0},...,w_{in})$ is the initial weight vector for the ith PE in that layer and $\mathbf{W}'_i = (w'_{i0},...,w'_{in})$ is the updated weight vector, $\mathbf{M_i} = (m_{i0},...,m_{in})$ is the accumulated weight changes for the ith PE and $\mathbf{A}_i = (a_{i0}, \ldots , a_{in})$ is the auxiliary weight field that is used as momentum term. Lcnt is the count of learned samples and AUX1(epoch size) is the accumulation period. The RMS error (RMSE) is the stopping criterion for training and is defined as

$$RMSE = \sqrt{\frac{1}{2}\sum_{i=1}^{AUX1} \| \mathbf{Y}^{(i)} - \mathbf{D}^{(i)} \|^2} \quad (19)$$

One of the problems with the cumulative delta rule is that the learning coefficient C_1 depends on the epoch size AUX1. As the size AUX1 increases, C_1 should get smaller, otherwise the accumulated weight changes will become too large and cause the learning to diverge. Normalized cumulative delta rule gets around this problem by dividing the accumulated delta weight by the square root of the epoch size before being applied. Moreover, examples in the training set were presented to the network randomly during the training to avoid the "learn one thing but forget others" problem.

During the learning process, different schedules are used for adjusting the learning rates (parameters C_1 and C_2) for the input, hidden and output layers respectively (Table 2). Point fields, e.g. the intervals between transition points increase exponentially, and the coefficient ratio (e.g. 0.5) defines an exponential decay of the C_1 and C_2 for the hidden and output layer, which is sampled at subsequent transition points (e.g. $C_1 = 0.5, 0.5/2, 0.5/2^2, \ldots$). The coefficients for the input layer are not changed with the learning iterations.

Table 2. **The learning schedule**

Schedule for the input layer					
LCNT	50000	0	0	0	0
C_1	0.9000	0.0000	0.0000	0.0000	0.0000
C_2	0.6000	0.0000	0.0000	0.0000	0.0000
Schedule for the hidden layer					
LCNT	10000	30000	70000	150000	310000
C_1	0.3000	0.1500	0.0375	0.0023	0.0000
C_2	0.8000	0.4000	0.1000	0.0063	0.0000
Schedule for the output layer					
LCNT	10000	30000	70000	150000	310000
C_1	0.1500	0.0750	0.0188	0.0012	0.0000
C_2	0.8000	0.4000	0.1000	0.0063	0.0000

B. *Road Classification*

First the rotation-invariant Fourier amplitudes are used to train the RCN. In this case the RCN has 16 inputs (a_1,...,a_{16}) and 6 outputs. Experiments indicated that 16 or 32 is the proper size of the update epoch AUX1. If the epoch is too small (e.g. 8), noise in the data set seems to confuse the network and the network may oscillate; On the other hand, if the epoch size is too large (e.g. 64), proper adjustments may be ignored and the network may not converge. The number of the hidden units , h, is decided by systematic experimental analysis in order to find the minimum number for the proper classification and best number for the problem. Table 3 shows the training results for different number of hidden units (h=0~16). "C" is the number of training iterations at which the network becomes stable. The RMS errors "e" are also listed in the table.

Table 3. The learning process of the RCN

h	0	3	4	5	6	7	10	**12**	16
C	53634	51137	86752	73906	65581	75280	71064	**86923**	50939
e	0.30	0.25	0.20	0.20	0.15	0.15	0.12	**0.10**	0.14

(h: Number of the hidden units; C: Training iterations; e: RMS error)

Each realization of the RCN was tested by using the training set, testing set and the original raw data set, which may include noisy data. If the value of one (e.g. kth) of the six network outputs $Y=(y_1, \ldots, y_6)$ is greater than 0.5 and the other five values are less than 0.5, then the input road image is thought of being correctly classified and is assigned to kth category. Table 4 lists the correct recognition rate (%) for the three data sets under every realization of the RCN.

Table 4. Recognition rates of the RCN using Fourier amplitudes

with various h, the number of hidden units

set \ h	0	3	**4**	5	6	7	10	**12**	16
training	86.9	91.8	**96.3**	94.6	96.7	96.6	97.6	**98.5**	98.5
testing	83.5	86.5	**91.6**	88.9	92.5	91.6	92.9	**94.4**	94.2
original	77.2	79.5	**86.1**	84.1	85.9	85.3	87.6	**89.0**	88.6

The training and testing process indicates that 4 is the minimum number of the hidden units for proper classification, and 12 is the best number. Comparing with the learning process of the networks using rotation-independent orientational projection as inputs in [10] , the network RCN converges much slower and the recognition rate is slightly lower, and more hidden units are needed for the best classification. The reason may be that the Fourier amplitudes lose the phase information of the orientational projection and the Fourier transform compacts the energy in the first several terms so that weights between the input and the hidden units are not balanced. However the rotation-invariance and fewer components of the Fourier amplitude vector makes itself a good choice for the input of the RCN.

We also try to use the auto-correlation function (ACF), which is calculated as the inverse Fourier transform of the energy spectrum of the orientational projection and is still rotation-invariant, as the inputs of the RCN. Although this transform does not add or lose any information, experiments indicate that this kind of data representation may be more distinctive for the BP network (refer to Fig. 13). The testing results using ACF show improvements in convergence speed (half the iteration number as using FA) and recognition rate while the number of hidden units is fewer (Table 5). Analysis of the weight patterns shows that the connections between the input and the hidden layers extract more distinctive features from this kind of input patterns, which are more suitable for road classifications. However the improvements of the RCN with input ACF require more computation power for inverse Fourier transform and processing more input units.

Table 5. Recognition rates of the RCN when input is ACF

set \ h	**3**	4	5	**6**	7	8
training	**95.24**	94.37	97.37	**97.00**	80.35	80.85
testing	**92.22**	90.25	92.99	**92.73**	90.38	92.67
original	**87.17**	84.71	87.17	**88.85**	82.74	84.67

C. Road Orientation Estimation

After the road category is determined, the corresponding RON for this category is activated. We compare the results of the networks using original orientational projection U and the Fourier phases Ψ as inputs, and having different number of units in the single hidden layers. The outputs of the RON are L(=32) discrete orientations. The three-layer RONs with different number of hidden units (from zero to sixteen) do not converge when the input is the phase data. The reason might be that the phase data is sensitive to noise and has additive ambiguous. So we use the original orientational projection U as the inputs of the RCN. Experiments using 0~16 hidden units in a single hidden layer show that the RONs with no hidden units (i.e., two-layer BP network) perform best for all the road categories. More detailed analysis will be given in the next sub-section. Table 6 shows the correct orientation estimation rate, measured by percentage of errorless estimation, and the average orientation error of the rest in percentage ($100 \mathrm{x} \Delta\phi/2\pi$), for each road category. The epoch size of learning process for each road category is also presented in Table 6. The estimation accuracy decreases and the epoch size should be large when the input data become noisy and scattered for a certain category (e.g., “D”). It also indicates that this category should be further divided into several sub-categories. The robustness of orientation estimation could be improved by integrating the RON result with the information of the orientation difference of the temporal sequences calculated by equation (13).

Table 6. Recognition rates and errors of the two-layered RONs with input U

class	||	T	+	D	C	S
epoch size	4	1	1	16	8	8
training set	100. (0)	99.9 (0)	100. (0)	83.9 (4)*	96.2 (2)	89.4 (1)
testing set	99.0 (0)	100. (0)	100. (0)	83.1 (4)	95.7 (2)	86.7 (1)
original set	96.8 (1)	98.1 (1)	87.9 (5)	70.4 (9)	90.8 (4)	76.5 (6)

*For example 83.9 (4) means the correct rate is 83.9% while the average error for the rest is 4%.

D. Internal Representation

A standard Y-layer feed-forward networks consists of an input layer, (Y-2) hidden layers and an output layer of units successively connected in a feed-forward fashion with no connections between units in the same layer and no feed-back connections between layers. The classical single-layer perceptron (two-layer BP network in our context), given two classes of patterns, attempts to find a linear decision boundary separating the two classes. If the two sets of patterns are linearly separable, the perceptron algorithm is guaranteed to find a separating hyperplane in a finite number of steps. It is commonly accepted that a single layer perceptron is inadequate for situations with multiple classes and nonlinear separating boundaries. Hence the multi-layer perceptron network(MLP) was proposed. The MLP net can be viewed as a nonlinear input-output mapping, and the learning process can be seen as fitting a function to the given data set.

A neural network is widely regarded as a black box that reveals little about its predictions. However, analysis of the road classification network RCN with 32 input units and 6 hidden units reveals some properties of the internal representation, especially the role of the hidden units. We developed the concept in [6] to define the receptive fields of hidden units as the distribution the connection weights from all the input units to each hidden units and the integrating fields as the distribution of the connection weights from all the hidden units to each of the output units. Roughly speaking each hidden unit extracts some kind of features from the input units through the corresponding receptive filed, and the integrating fields integrate several features to conclude the final recognition. It is difficult to describe the mechanism clearly and needs further study. Recent works[18,19] show that rules can be extracted from ANNs. In our experiments we found that the most of the outputs of the hidden units are saturated (near + or near -1, see Fig. 14) . Since we use TanH as the transfer function, it means that some kinds of features are detected (+1) or not detected (-1). The unstable outputs (with absolute values less than 0.5) of the hidden units means that the network is not certain about those features.

c\h	*1*	*2*	*3*	*4*	*5*	*6*
||	+	$^{+}$	--	+	+	--
T	--	-	--	--	+	+
+	+	+	--	--	--	-
D	+	--	+	+	+	--
C	+	$^{+}$	+	+	+	+
S	+	--	+	+	--	--

Fig. 14. The output of the hidden units (+: around 1.0 ;--:around -1.0;
+ : around 0.5 ; - : around -0.5; c: category; h: No of the hidden unit)

Experiments also show that some of the input units are not very important so that pruning them does not effect the recognition results very much. Similarly, some of the hidden units (features), for example the second hidden unit, are not vital for the road classification. We found that the RCN, virtually a distributed processing system, still work in case of the disability or the injury of some parts of the system. The physical meaning of the disability of the input units may be the partial occlusion of the scene by other objects, the partial changes of the environments, or the injury of the “eyes”. Similarly the disability of the hidden units means the injury of some part of the “brain”. The disability of more important inputs and/or hidden units (e.g., the third one) have more negative effects to the performance of the network, but effects are only severe to some of the categories (e.g. ,“S”) .

Analysis of the RONs weights reveals that the operations of the two layered orientation networks are virtually correlation functions. The Q-dimension weight vector between Q input units and the kth output unit is nearly a circular k-shift version of the representative vector of the input for the given class of the RON. The weight vector corresponding to each output unit is compared with the input vector. The best match indicates the correct orientation of the input vector. In other words, the circular-shifted version of the orientational projection for a given road class are almost linear separable.

The reason why we use RONs to estimate the road orientation instead of direct correlation is that the neural networks can learn the best templates (the representative vector) from the training examples.

E. Practical Considerations

ANNs are essentially massive parallel computing systems consisting of an extremely large number of simple processors with many interconnections. State-of-the-art computer technology such as VLSI and optics has made this possible. The computation requirements of road scene understanding using the RUNN in the current available serial computer consist of the following four steps.

1. Polar transformation and projection in the PIPE. The time complexity is O(RN) where R is the radius dimension and N is the orientation dimension of the polar OVI. These two geometrical transformations can be realized in the PIPE in real-time when R and N are 80 and 360 respectively.

2. 1D Fourier transform in PC. The computation complexity is $O(N\log_2 N)$ using 1D fast Fourier transform where N is the number of the orientations. This takes about 1.0 second in PC system (CPU 486/66M Hz) when N = 360 and takes less than 0.1 second when N = 32.

3. Road classification using the RCN. The time complexity is O(hP+hM) where P, h and M are the number of units in the input layer, hidden layer and output layer of the RCN respectively. This number is 424 when P=16, h=12 and M =6 and it takes about 0.1 second from the input to the output by using Nworks.

4. Road orientation estimation using the RON. The time complexity is O(N^2) where N is both the number of units in the input layer and output layer of the RON. This number is 1024 when N=32 and it takes about 0.2 second from the input to the output by using Nworks.

In the current experiments of the RUNN training and testing, the orientational projection is re-sampled from 360 to 32 since the software Nworks running on a PC486 is much slow with large number of PEs in the training process. Correspondingly the number of the outputs of the RON, the estimated orientations, is sampled to 32 in our principle experimental study. It means the angle resolution is about 11 degrees for the 360-degree view, which can not meet the practical scene requirements. This problem could be solved by using the N=360 inputs of original orientational projection to the RONs. Experiments with 360-sampled orientational projections shows that the orientation difference given by the cross-correlation-function (CCF) method is the average of the orientation difference between the front roads and the orientation difference between the back roads in the two images (Fig. 8). It means that acceptable results can be produced by the correlation approach. Since the RON acts as a correlation operation, the trained RON with high angular resolution should be better than the fixed CCF method.

Since the energy of Fourier spectrum compacts to the first few items, so the number of the inputs of the RCN , P, is much smaller than N. Experiments show that P=30 is enough to represent each category when N is 360 (refer to Fig. 13), so the scale of the RCN does not increase with N if we used Fourier amplitudes for road classification. Even if the ACF is used, the number of the input data is N/2 since ACF is symmetric about N/2. However if we expect one-degree angular resolution for the orientation estimation, the number of both input and output units, Q and L, should be 360. It means there are N^2 (about 130K if N=360) connections and weights for each of the RONs. In the initial stage of the operation or the recovery stage when the robot has missed the road, the system should search for all the 360 directions. Fortunately in the normal conditions of continuous road following, the system only needs to search for a narrow range of orientations, for example, from -16 degrees to 16 degrees with one-degree interval. So only 33 outputs and the corresponding connections are activated, and most of the outputs and the corresponding connections could be disabled. Therefore even the RUNN is simulated in the general von Neumann computer, the real-time computation is possible for real applications if a faster PC system (i.e. Pentium II/266M Hz) is used.

VII. CONCLUSION AND DISCUSSION

In this paper we present the experimental results of training and testing the backpropagation network for the outdoor road scene understanding using omni-view images. Both the road orientations used for robot heading and the road categories used for robot localization are determined through the integration of invariant image analysis and adaptive neural networks. Several design issues, including the network model, the selection of input data, the number of the hidden units and the learning problems are studied. The internal representations of the networks are carefully analyzed, which could guide the further researches and applications. Experimental results with real scene images are promising. In order to actually apply the neural networks to the real world autonomous robot navigation in the outdoor natural scene environment, the following aspects are in consideration.

A. Using 2D Image Patterns

Keeping the requirement of extracting rotation invariant image features in mind, and at the same time reducing the dimension of the original 2D omni-view images, we are investigating the possibility of using the principal component analysis (the KL transform) along the radius axis r of the omni-view polar image I(θ,r) to obtain eigenfeatures for a given θ

$$\mathbf{U}(\theta) = (u_0(\theta), u_1(\theta), \ldots, u_S(\theta)) \quad (20)$$

where S is expected to be much smaller than the original dimension of r. Preliminary experiments show that the first 3 components of KL coefficients of the 1D radius image can properly represent the original 2D omni-view image with r from 0 to 127. The sampled eigenfeature sequence along the orientation direction can be expressed as

$$\mathrm{U}=\{\, \mathbf{U}(n) = (u_0(n), u_1(n), \ldots, u_S(n)),\ n = 0, \ldots, N-1 \,\} \qquad (21)$$

Sequence $\{u_0(n)\}$, the first component of $\mathbf{U}(n)$, is approximately the orientational projection define in this paper. The rotation-invariant vector sequence can be obtained by applying the 1D discrete Fourier transform(DFT) to each component sequence. The computation complexity for 1D KLT-DFT method is only O($SN(R+\log_2 N)$) where S is the dimension of the eigenfeature, N is the number of orientations and R is the radius of the polar image .

B. Using Temporal Coherence

Spatio-temporal pattern recognition (SPR) network was proposed by S. Grossberg[20,21] to explain certain cognitive process for recognizing sequence of events. The primary application of

SPR networks appears to be in the area of recognizing repetitive audio signals. It is straightforward to apply the SPR network for recognizing image sequence of outdoor road scene. Since the same road category will last for a period of time, SPR network should not be sensitive to the occasional image events and could give a robust recognition.

C. Self-Organization and Unsupervised Learning

The natural extension of the work is to use the unsupervised self-organization neural network[22]. When the robot has enough ability to travel around the known world by itself, we can expect that it can also explore the unknown world by itself.

D. Multiple Sensor Fusion and Integration

Visual navigation of a mobile robot in the natural environment is a difficult and comprehensive subject, which is related to almost every aspects of computer vision researches. The fundamental tasks of visual navigation are composed of global localization, road following and obstacle detection. Environment modeling is the foundation of visual navigation. A task-oriented, multi-scale and full-view scene modeling strategy is proposed for visual navigation in natural environment [12]. It combines the panoramic vision for scene modeling, omni-directional vision for road understanding and binocular vision for obstacle detection into an integrated system. This approach overcomes the drawbacks of traditional visual navigation methods that mainly depended on local and/or single view visual information.

Acknowledgments

This work was supported by the Advanced Research Project of China. We are grateful to the anonymous reviewers of this paper for their valuable comments and suggestions.

References

[1] Ron T. Elkins and E. L. Hall, Three dimensional line following using omnidirectional vision, *Proc. SPIE* vol. 2354 , 1994: pp 130-144

[1] Y. Yagi, S. Kawato, S. Tsuji, Real-time omnidirectional image sensor (COPIS) for vision-guided navigation, *IEEE Trans Robotics and Automation*, vol 1, no 1, Feb 1994 :pp 11-27

[2] J. Hong, X. Tan, B. Pinette, R. Weiss, E. M. Riseman, Image-based homing, *Proc Int Conf Robotics and Automation*, April 1991, pp 620-625.

[3] F.Stein and G. Medioni, Map-based localization using the panoramic horizon, *Proc Int Conf Robotics and Automation*, 1992

[4] Y. T. Zhou, R.A. Chellappa, A network for motion perception, *Proc. IJCNN* , 1990: pp II841-851.

[5] E. Atsumi, et al, Internal representation of a neural network that detects local motion , *Proc. IJCNN, 1993*: 198-201.

[6] D.A. Pomerleau, Neural network based autonomous navigation, *In Vision and Navigation : the CMU Navlab*, Kluwer Academic Publishers, Boston, 1990.

[7] T. Jochem, *Using virtual active tools to improve autonomous driving tasks*, Technical Report, CMU, Nov.1993.

[8] T. Jochem, D.A. Pomerleau, Thorpe C, Vision guided lane transition, *IEEE Sym on Intelligent Vehicles*, Detroit, USA, Sep 25-26,1995

[9] Z.G. Zhu, G.Y. Xu, Neural networks for omni-view road image understanding , *J. of Comput. Sci. and Technol.*, vol 11, no 4, July 1996.

[10] Z.G. Zhu, H.J. Xi, G.Y. Xu, Combining rotation-invariance images and neural networks for road scene understanding，*Proc IEEE ICNN,* 1996: pp 1732-1737.

[11] Z.G. Zhu, *Environment modeling for .visual navigation*, PhD Dissertation, Tsinghua University, May 1997

[12] D.E. Rumelhart, B. Widrow, M..A. Lehr, “ The basic ideas in neural networks,”, *Comm. ACM*, Mar 1994, vol. 37, no 3 : pp87-91

[13] R. Linggard, et al (ed.), *Neural Networks for Vision, Speech and Natural Language,* Chapman & Hall , First Edition, 1992.

[14] A.K Jain, J. Mao, K.M. Mohiuddin, Artificial neural networks: a tutorial, *IEEE Computer,* vol 29, no 3, March 1996:pp31-44.

[15] Neuralware Inc., *Neural Computing: Using Nworks of Professional II/Plus*, 1991.

[16] Aspex Incorporation, *The PIPE Reference Manual* , 1989.

[17] G.G. Towell, J.W. Sharilik, Extracting refined rules from knowledge-based neural networks, *Machine Learning,* vol 13, no 1, Oct. 1993: pp71-101.

[18] R. Setiono, H. Liu, Symbolic representation of neural networks, *IEEE Computer*, vol 29, no 3, March 1996:pp71-78.

[19] GrossBerg, S., Some networks that can learn, remember and reproduce any number of complicated space-time pattern, I. *J. Math. & Mech.* , vol. 19, 1969: pp53-91

[20] GrossBerg, S., Some networks that can learn, remember and reproduce any number of complicated space-time pattern, II. *Studies in Applied Mathematics*, vol. 49, 1970: pp 135-166.

[21] Kohonen, T., *Self-Organization and Associative Memory*, Springer-Verlag, Berlin, 1984.