ALVINN

Alex Waibel

Interactive Systems Laboratories
Carnegie Mellon University
University of Karlsruhe

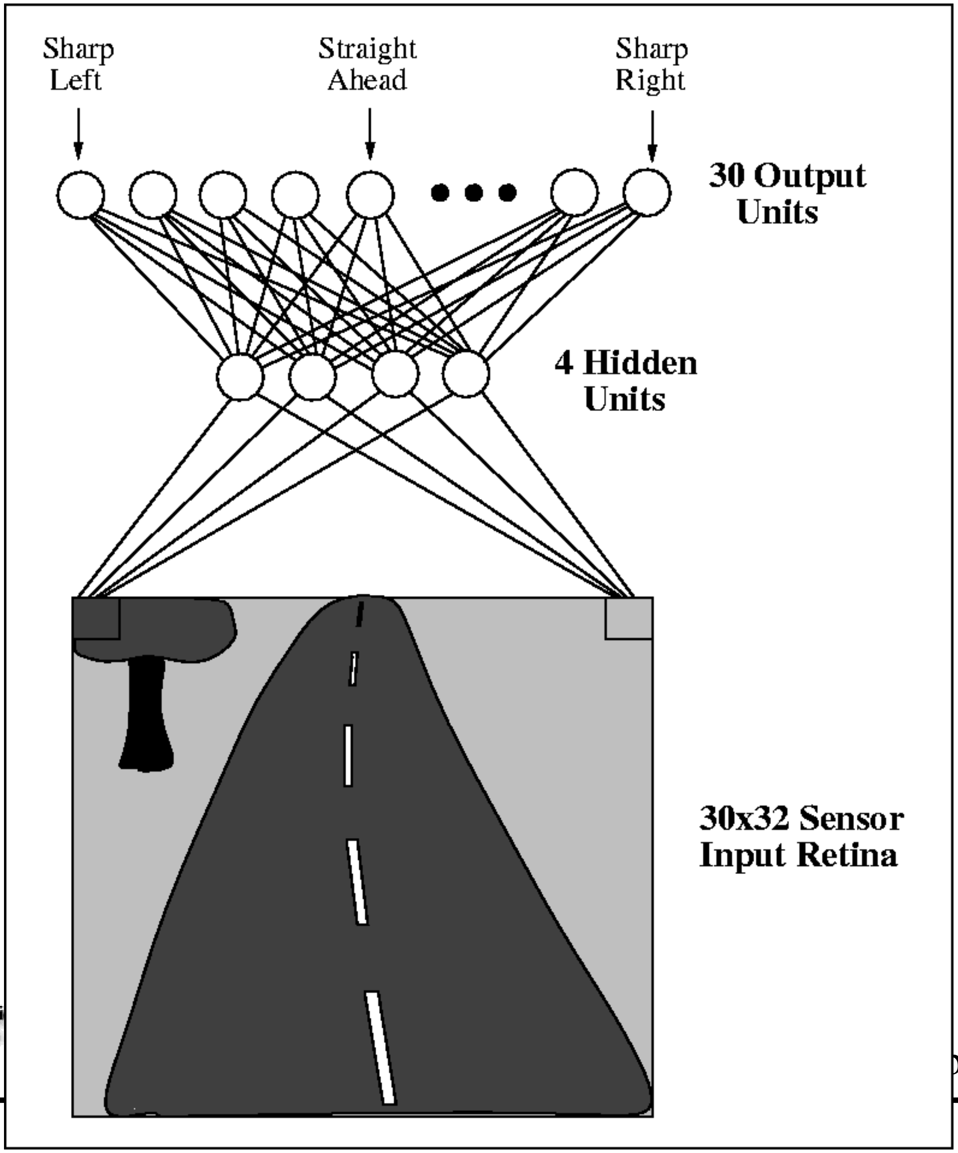
http://www.is.cs.cmu.edu

Email: waibel@cs.cmu.edu

ALVINN: Autonomous Land Vehicle In a Neural Network

- •Dean Pomerleau's Ph.D. thesis (1992).
- How ALVINN Works
  - Architecture
  - Training Procedure
  - Performance
- Why ALVINN Works
  - Hidden Unit Analysis
- Integrating Multiple Networks
- Other Applications

ALVINN Network Architecture



How many inputs?
$30 \times 32 = 960$

How many weights?
$961 \times 4 + 5 \times 30 = 3994$

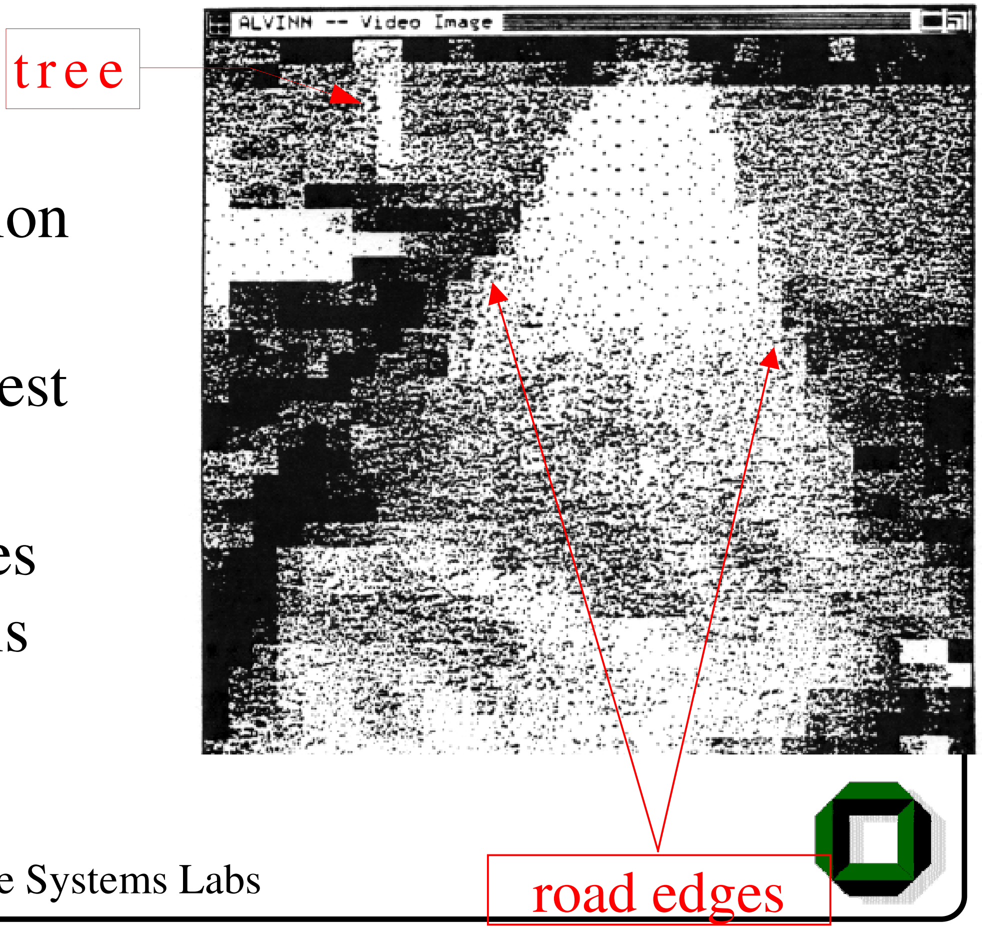
Original Training Scheme

•Generate artificial road images mimicing situations the network is expected to encounter, including noise.

•Calculate correct steering direction for each image.

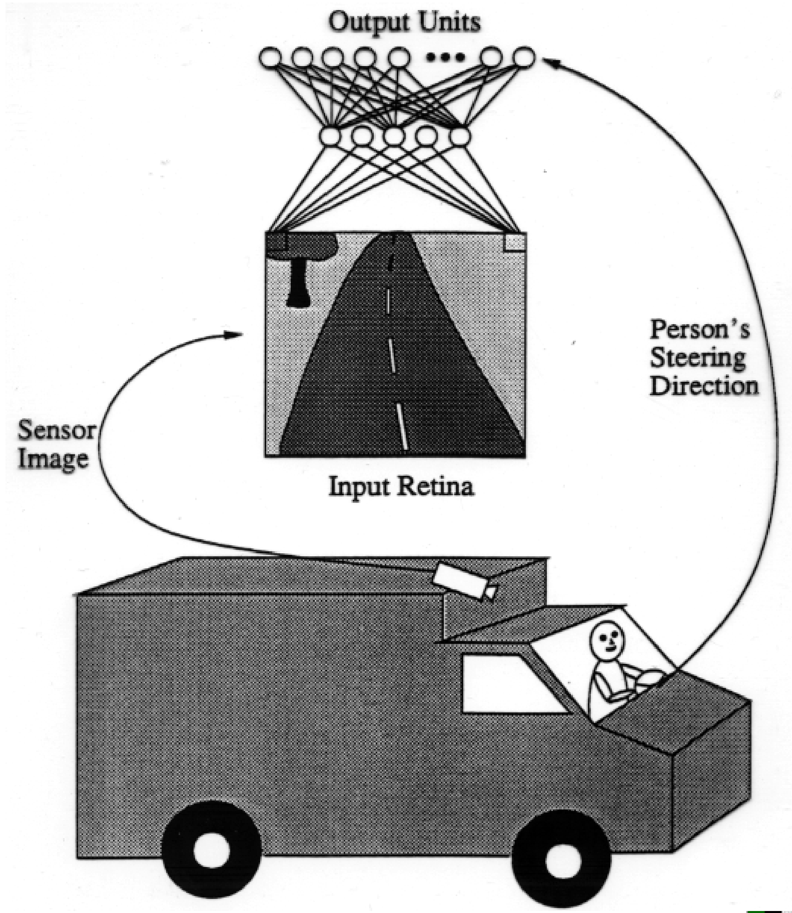
•Train on artificial images, then test on real roads.

•Problem: realistic training images are difficult to produce: training is expensive.



Training on the Fly

- Digitize the steering wheel position.
- Train the network by having it observe live sensor data as a human drives the vehicle.
- The human "teaches" the network how to drive.
- Can this really work?
    - It's not so simple...

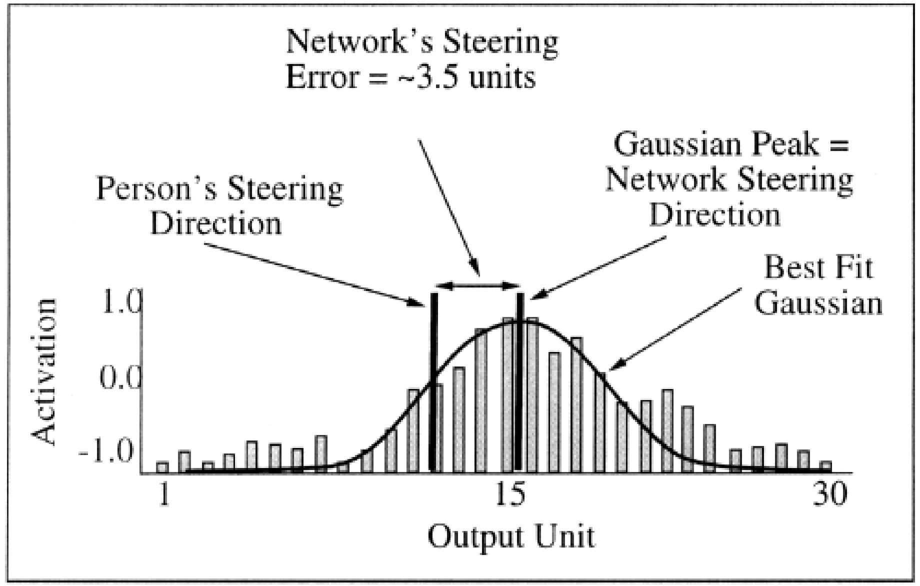


Measuring Steering Error

- Train with a Gaussian bump centered over the desired steering direction.

- To test: fit a Gaussian to the network's output vector.

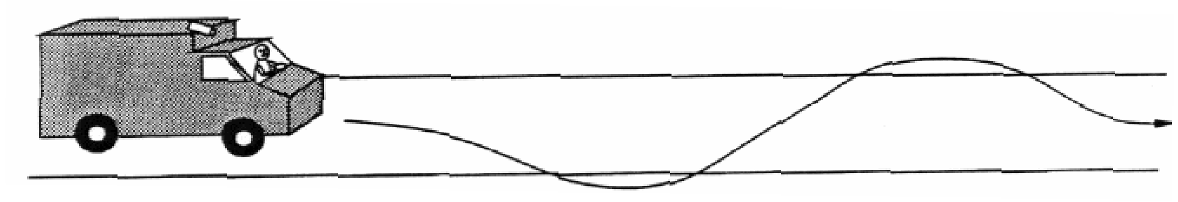
- Measure distance between Gaussian's peak and human steering direction.



Why use a Gaussian for the output pattern?

Learning to Correct Steering Errors

- If the human drives perfectly, the network never learns to make corrections when it drifts off the desired track.
- Crude solution:
  - –Turn learning off temporarily, and drive off course.
  - –Turn learning back on, and let the network observe the human making the necessary corrections.
  - – Repeat.



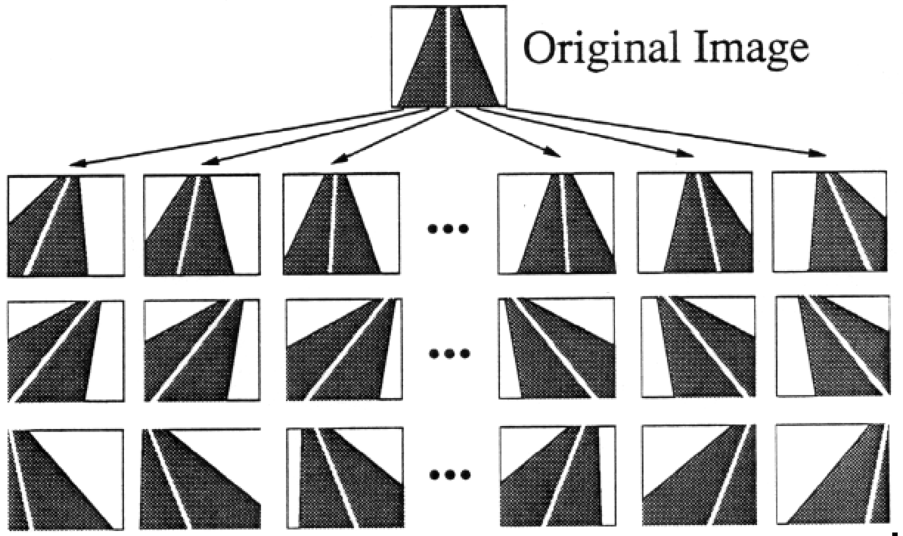
- Relies on the human driver to generate a rich set of steering errors: time consuming and unreliable.

Can be dangerous if training in traffic.

Simulating the Steering Errors

•Let humans drive as best they can.

•Increase training set variety by *artificially* shifting and rotating the video images, so that the vehicle appears at different orientations relative to the road.

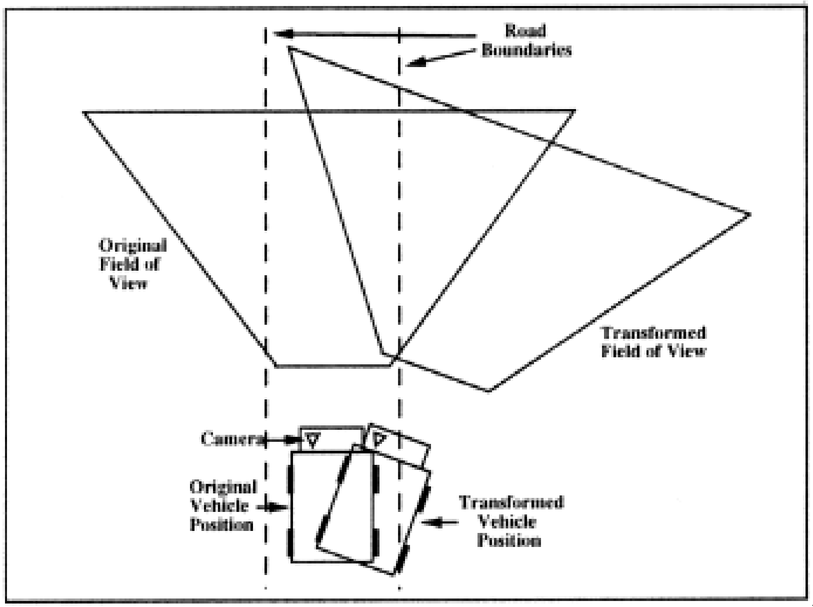


•Generate 14 random shift/rotations for each image.

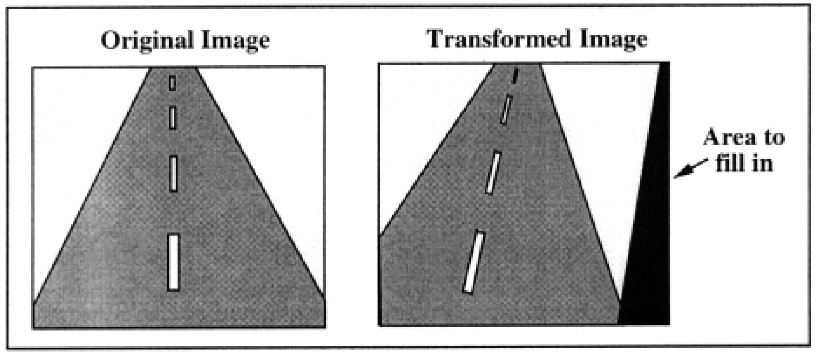
•A simple steering model is used to predict how a human driver would react to each transformation.

Road Shifts Lead to Missing Pixels

•Rotating and translating the camera can be simulated by copying pixels. But what about pixels in the new field of view that weren't present in the original camera image?


Road Boundaries
Original Field of View
Transformed Field of View
Camera
Original Vehicle Position
Transformed Vehicle Position

An exaggerated rotation



Effects of a modest shift

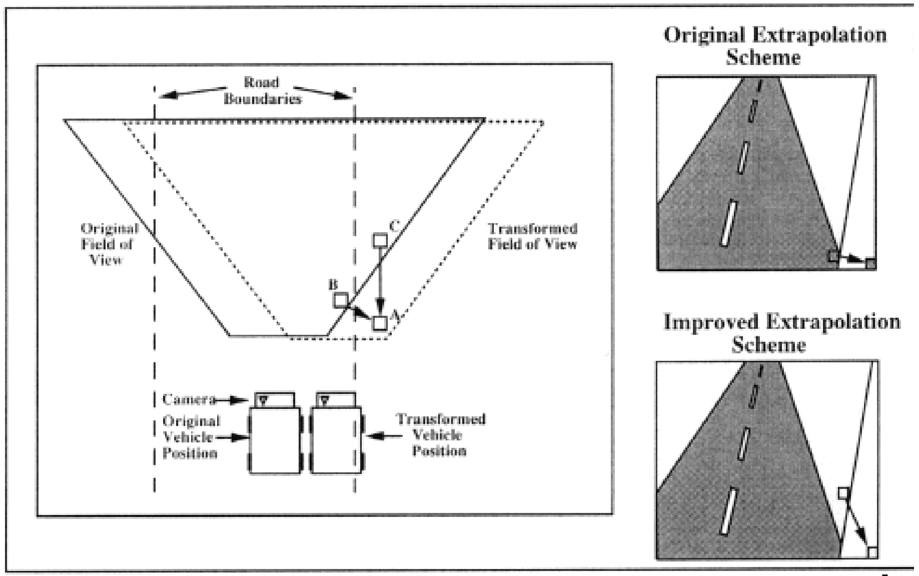
Filling In Missing Pixels

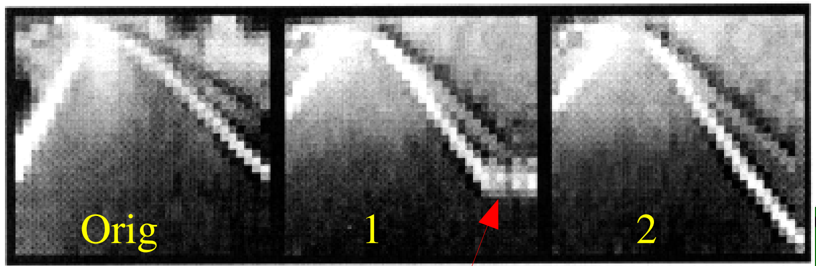
•1. Fill in (A) from closest image pixel (B).

•Problem: smearing.

•The smear becomes an image "feature" that the network learns to exploit!

•2. Project along a line from (A) parallel to the vehicle's heading to find closest pixel (C).

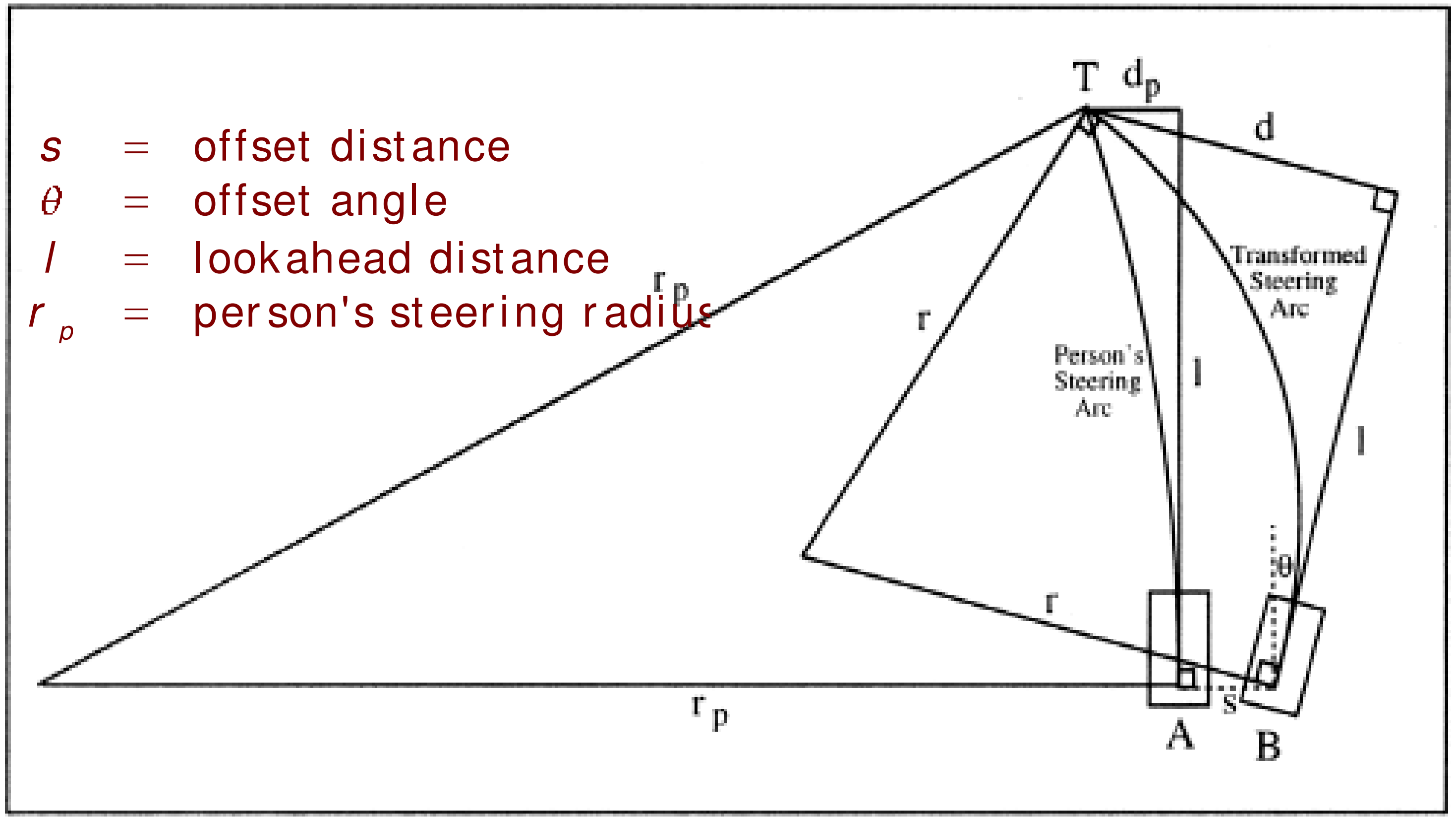

Road Boundaries
Original Field of View
Transformed Field of View
C
B
A
Camera
Original Vehicle Position
Transformed Vehicle Position
Original Extrapolation Scheme
Improved Extrapolation Scheme



smearing

Estimating Correct Steering Direction

•"Pure pursuit" steering model generates a fairly good estimate of what the human driver would do.

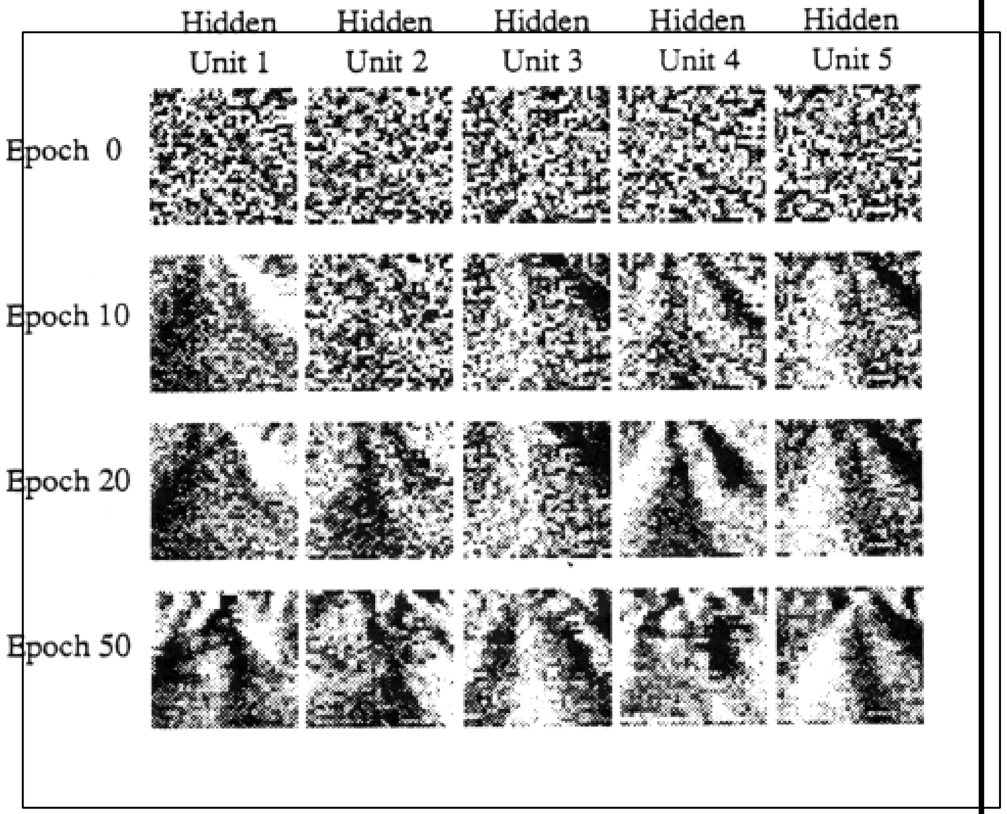


s	=	offset distance
θ	=	offset angle
l	=	lookahead distance
r_p	=	person's steering radius

Network Weights Evolving

•Initial random weights look like "salt and pepper" noise.

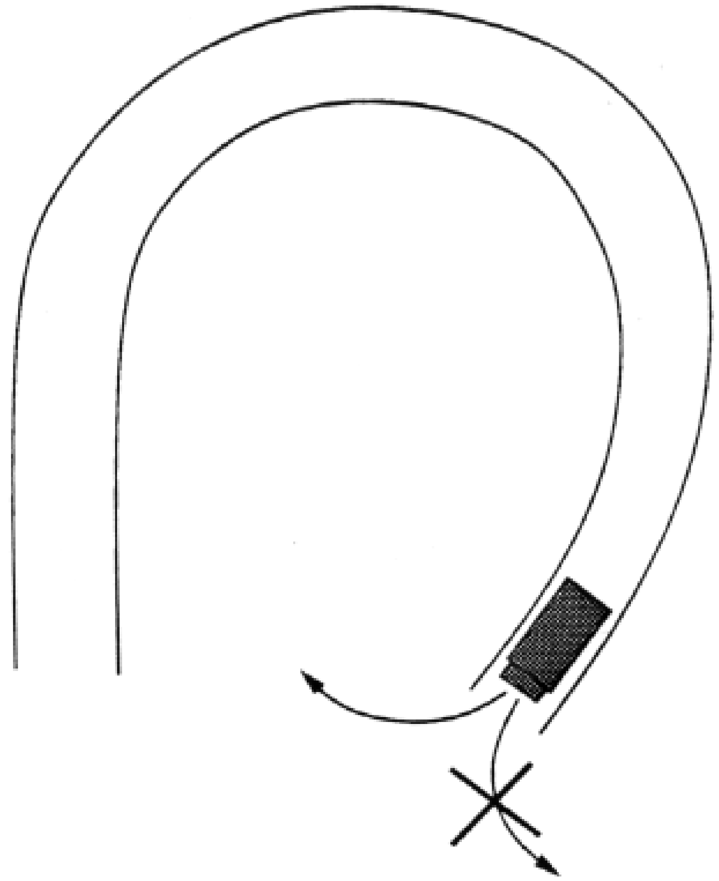
•During training, the hidden units evolve into a set of complementary feature detectors.



Problem with Online Learning: Network Can “Forget”

•The network tends to overlearn recently encountered examples and forget how to drive in situations encountered earlier in training.

•After a long right turn, the network will be biased toward turning right, since recent training data focused on right turns.



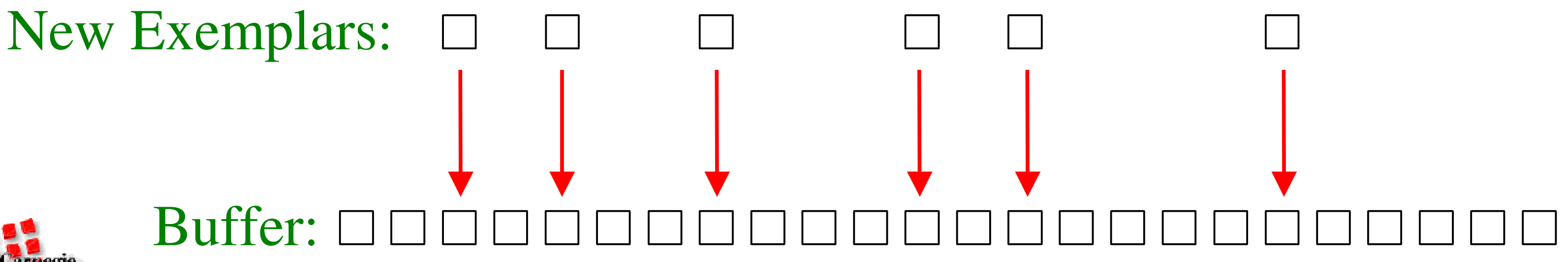
Solution: Maintain a Buffer of Balanced Training Images

This is a semi-batch learning approach. Keep a buffer of 200 training images.

Replace 15 old exemplars with new ones derived from the current camera image. Replacement strategies:

(1) Replace the image with the lowest error

(2) Replace the image with the closest steering direction



"Training on the Fly" Details

1) Take current camera image plus 14 shifted/rotated variants, each with computed steering direction.

2) Replace 15 old exemplars in the 200 element training exemplar buffer with these 15 new ones.

3) Perform one epoch of backpropagation learning on the training exemplar buffer.

4) Repeat steps 1-3 until the network's predicted steering direction reliably matches the person's steering direction.

5)How long does training take?

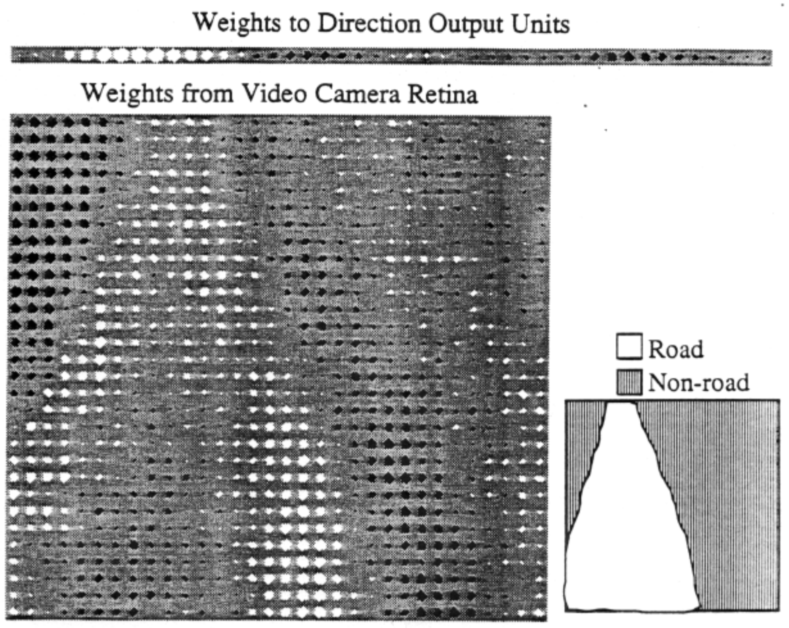
6) Just a few minutes!

(movie)

ALVINN Weight Diagram

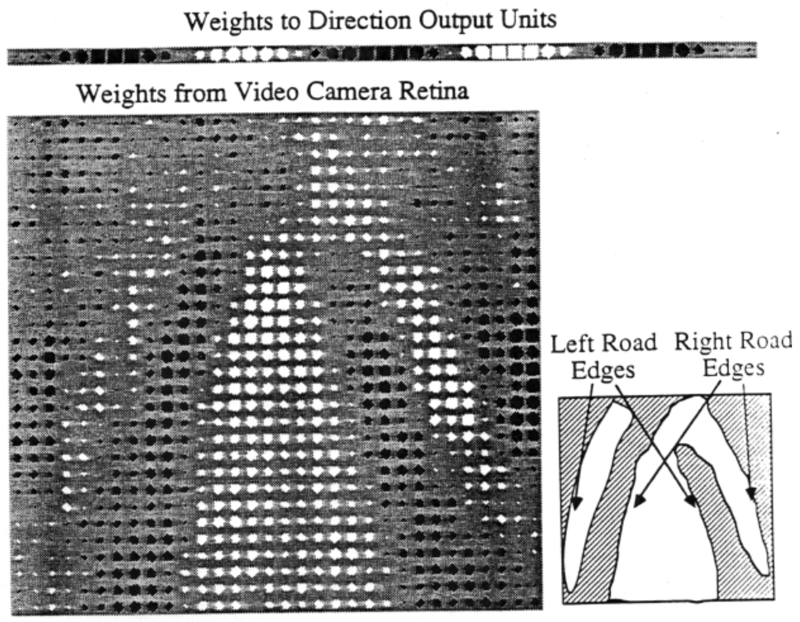
•This hidden unit is excited by a road on the left of the image.

•Its projections to the output layer are voting for a left turn, to bring the vehicle back to the center of the road.


Weights to Direction Output Units
Weights from Video Camera Retina
□ Road
▨ Non-road

ALVINN Weight Diagram

- This hidden unit is excited by roads slightly left OR slightly right of center.
- It suggests two steering directions: a shallow left turn and a shallow right turn.
- In order to determine which is correct, the network must combine outputs from several active hidden units.



Weights to Direction Output Units
Weights from Video Camera Retina
Left Road Edges
Right Road Edges

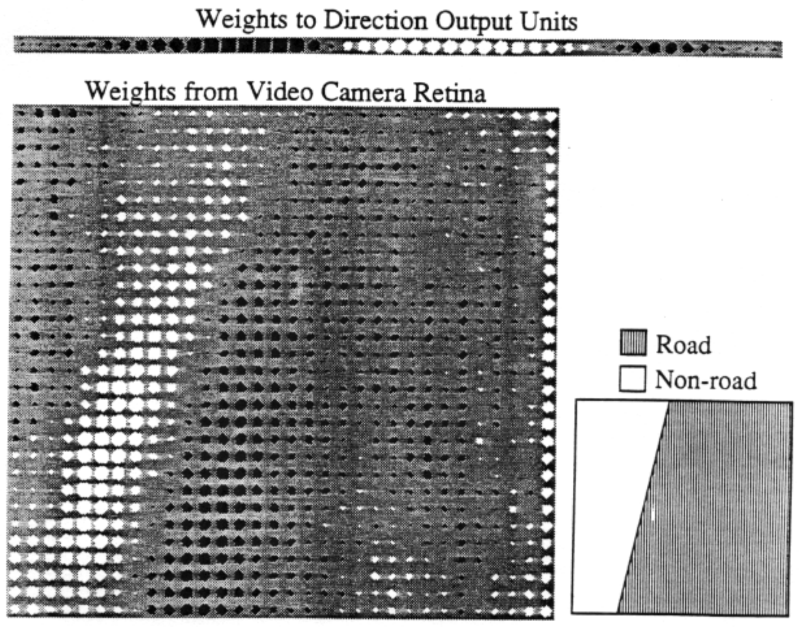
A "distributed representation" allows a very simple network to drive accurately.

ALVINN Weight Diagram

•This unit was taken from a network trained on a road whose width varied.

•In this case, the hidden units focus on detecting just one road edge.

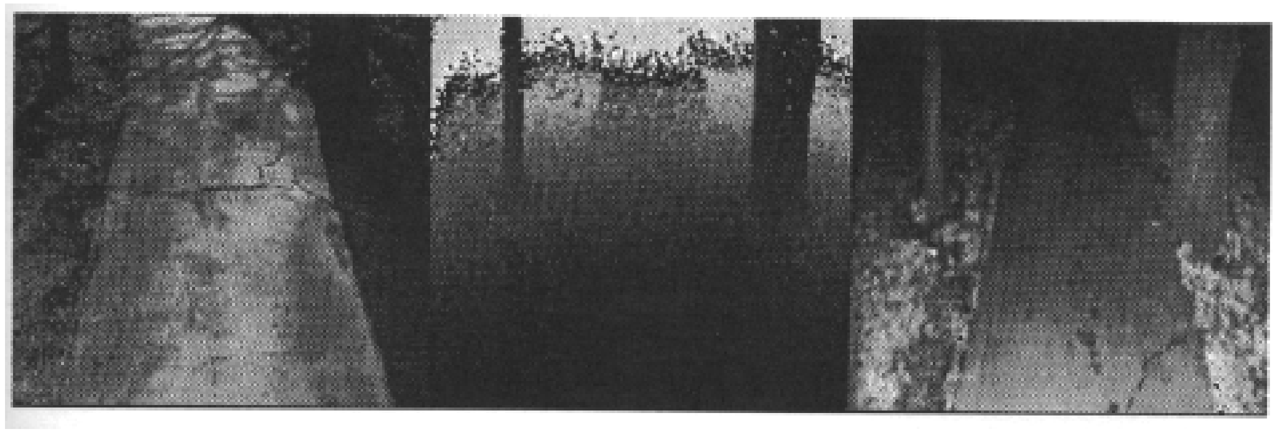
•The units vote for a relatively wide range of steering directions; their cooperative activity fine-tunes the steering direction.



Weights to Direction Output Units
Weights from Video Camera Retina
▨ Road
☐ Non-road

Multi-Modal Inputs

- ALVINN can avoid obstacles using a laser rangefinder. It can drive at night using laser reflectance imaging.



Regular Video

Laser Rangefinder

Laser Reflectance

Comparison with the "Traditional Approach"

1) Determine which image features are important, e.g., a yellow stripe down the center of the road.

ALVINN finds the important features itself.

2) Hand-code algorithms to find the important features, e.g., edge detection to find yellow lines.

ALVINN constructs its own feature detectors.

3) Hand-code algorithm to determine steering direction based on feature positions in the image.

ALVINN learns the mapping from feature detector outputs to steering direction.

ALVINN's Shortcomings

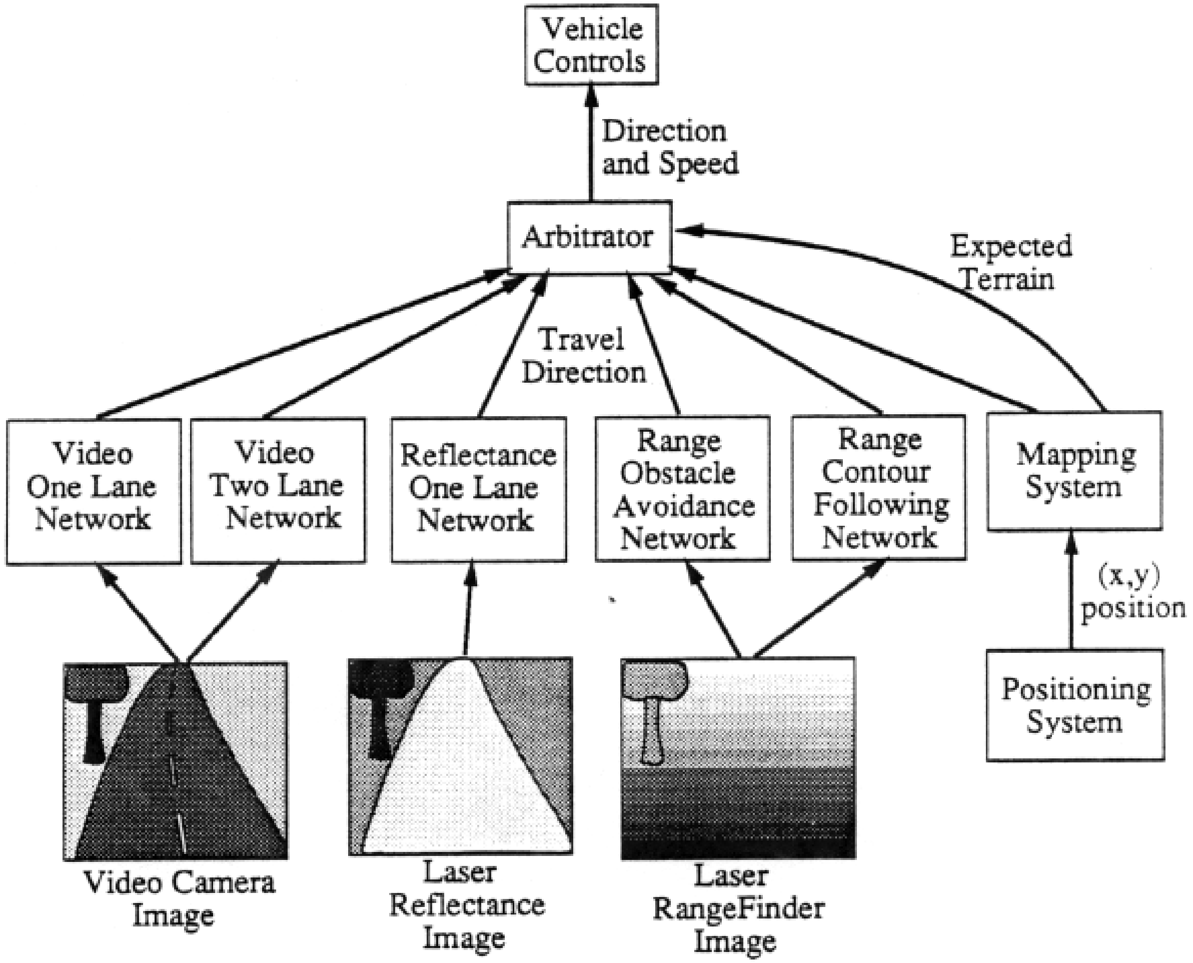
•The single-network ALVINN architecture can only drive on one type of road (unpaved, single-lane, double-lane, lane-striped, etc.)

•Can't transition from one road type to another.

•Can't follow a route.

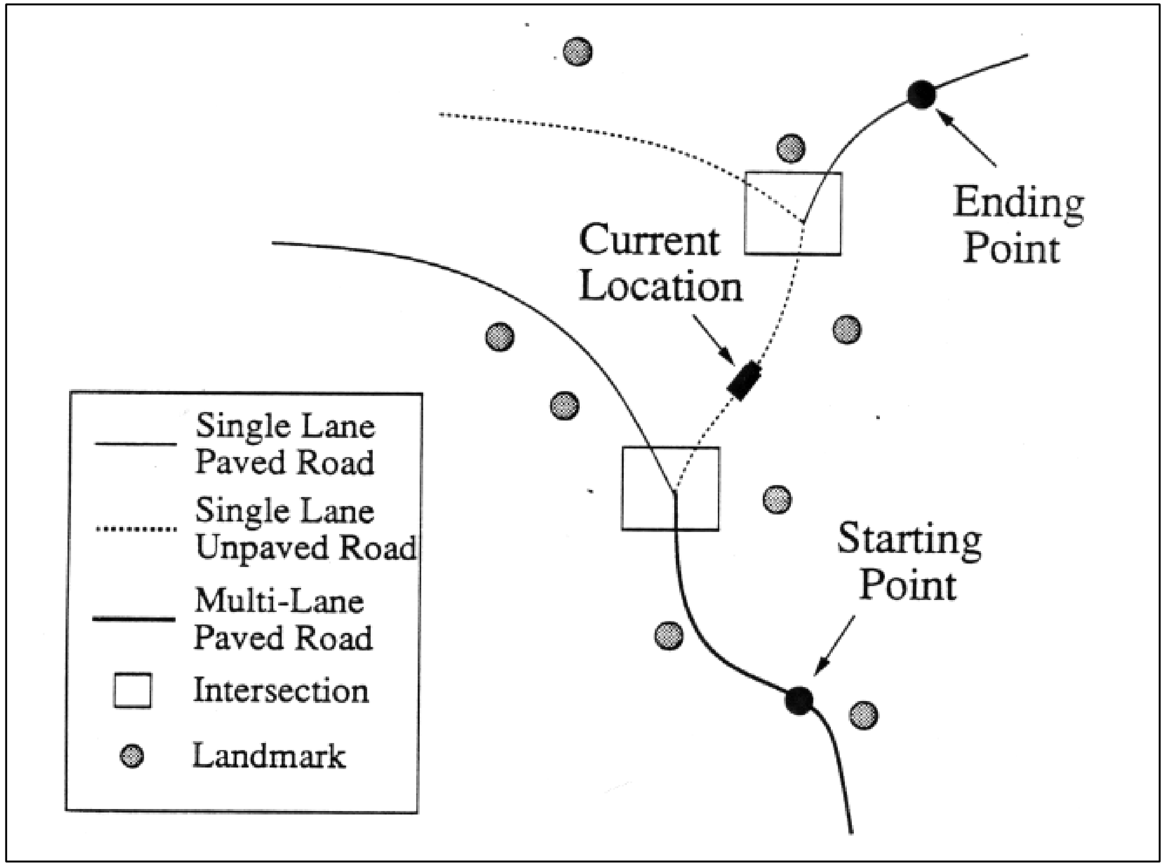
•Solution: rule-based multi-network integration.

Hybrid ALVINN Architecture



Symbolic Mapping Module

- Provides two kinds of information: (1) current road type, and (2) estimated steering direction.



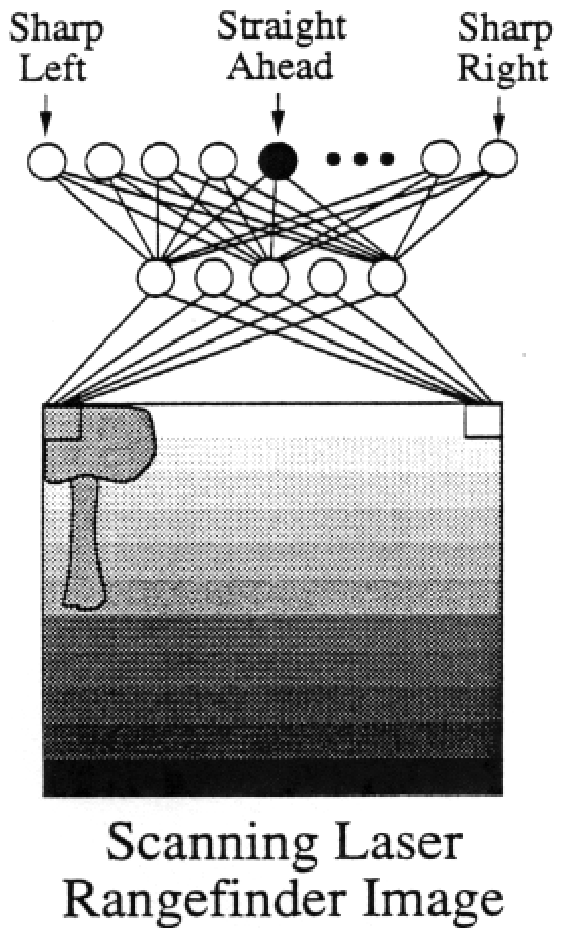
Map-Based Relevancy Arbitration

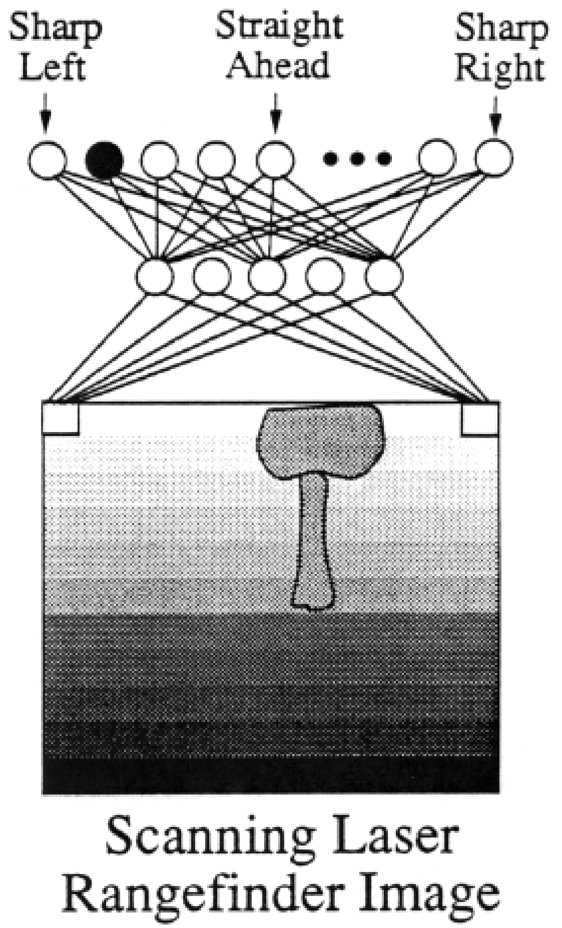
•Which module should drive the vehicle?

•If the map says the vehicle is on a two-lane road, the arbitrator will choose to listen to the two-lane road driving network.

•If the map says the vehicle is approaching an intersection, the arbitrator will choose to steer in the direction dictated by the mapping module in order to follow the correct path to the destination.

Obstacle Avoidance Network





Priority Arbitration

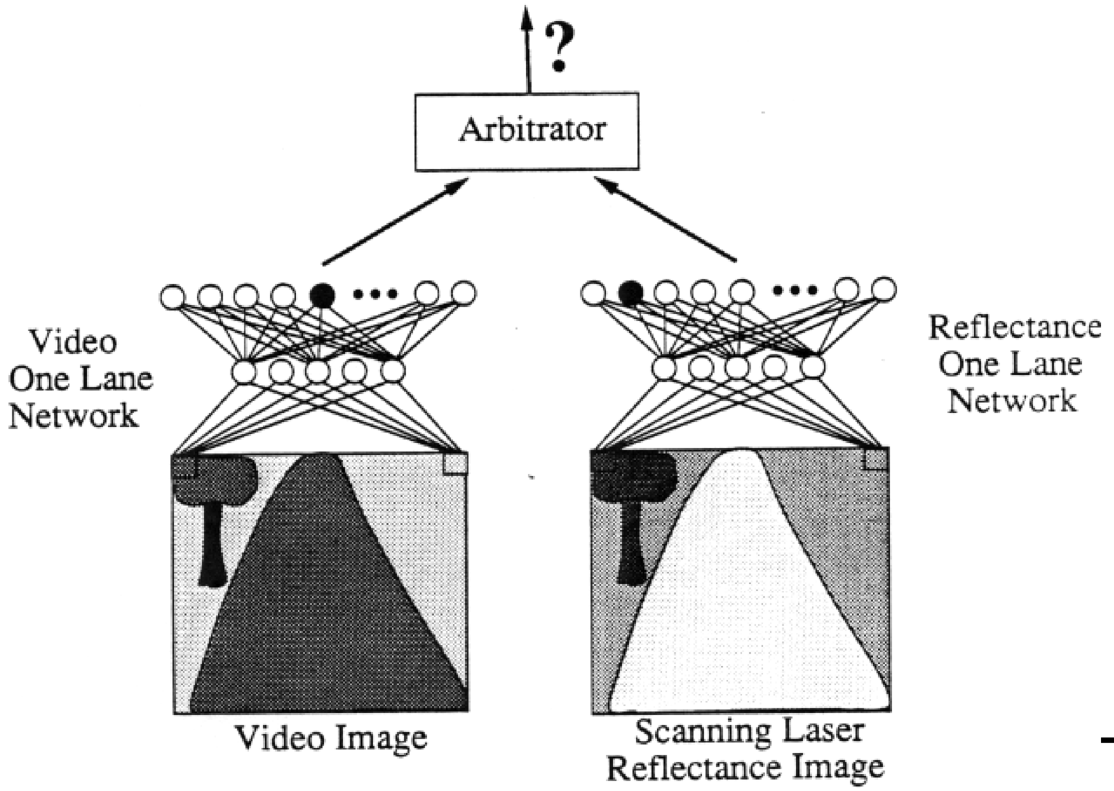
•The obstacle avoidance network is trained to steer straight ahead unless there is an obstacle in the way.

•The arbitrator will ignore the obstacle avoidance network when it says "Steer straight ahead," since the message has low urgency.

•But when the obstacle avoidance network suggests a sharp turn, its urgency is high, and the arbitrator will give it precedence over other behaviors.

Problems with Rule-Based Arbitration

•Relevancy and priority arbitration each choose a single network to listen to. What if we want to combine results from multiple networks?



Problems with Rule-Based Arbitration

•Requires detailed knowledge of each network's areas of expertise.

•Requires a detailed and accurate map of the environment.

•Assumes precise knowledge of the vehicle's position.

•Would like to be be able to say: “Go about ¼ mile and turn right at the intersection.”

Connectionist Arbitration

•Estimate the reliability of all networks in the current situation.

•Use the reliability estimate to:

1) Weight the outputs of the networks.
2) Pinpoint the vehicle's location.
3) Control the vehicle's speed.
4) Determine the need for retraining.

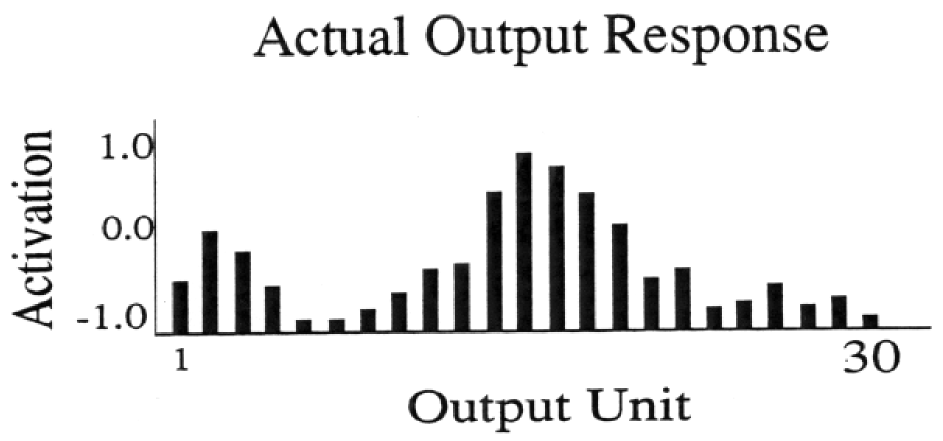
1)But how do you estimate a network's reliability?

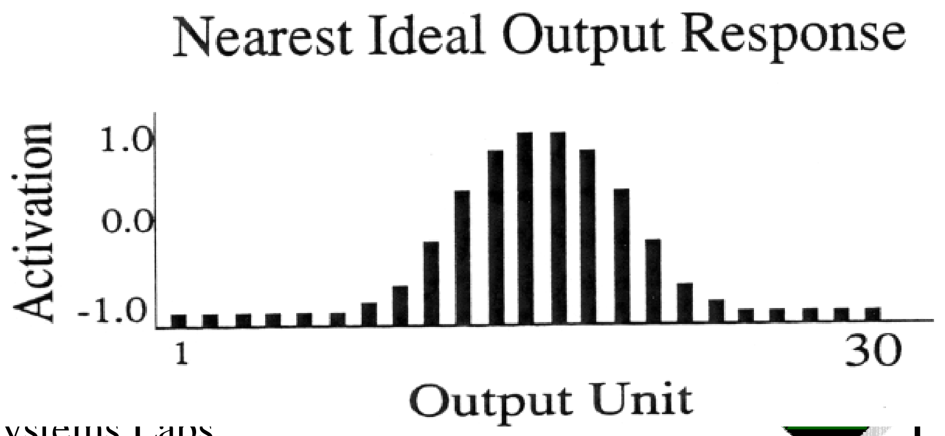
Output Appearance Reliability Estimation (OARE)

- Compare actual network output with the closest "ideal" output pattern.

•Calculate the "output appearance error".

•The larger the appearance error, the less reliable is the network.





Measuring Output Appearance Error

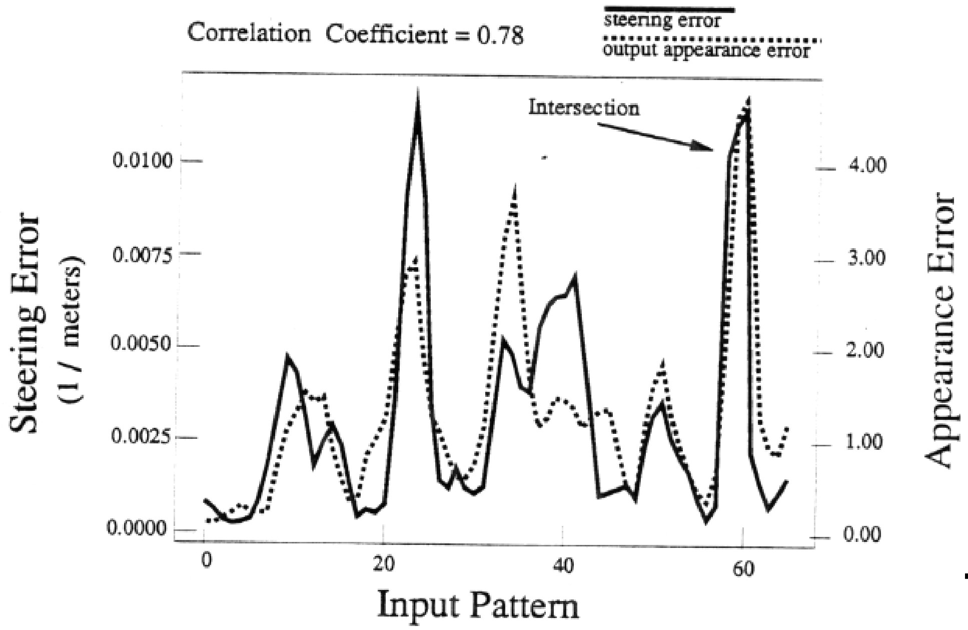
$$\text{Output Appearance Error} = \sum_i \left(\mathit{Actual}_i - \mathit{Ideal}_i \right.$$

$$\text{Steering Error} = \left| \mathit{curve}_h - \mathit{curve}_n \right|$$

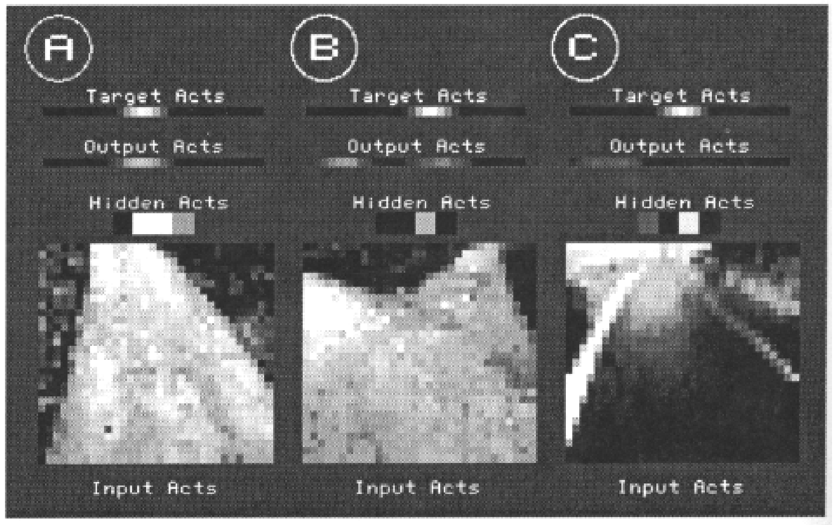
where $\mathit{curve}_h = \text{human turn curvature}$

and $\mathit{curve}_n = \text{network turn curvature}$

OARE predicts steering error. Correlation coeff. = 0.78



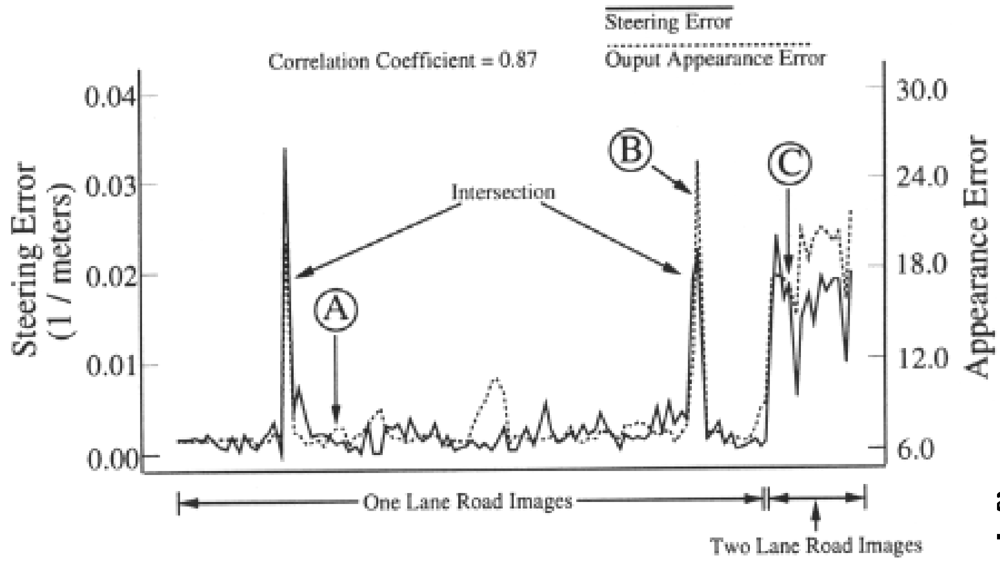
OARE Predicts Steering Error



A = one lane road (trained)

B = fork in road: output has bimodal distribution

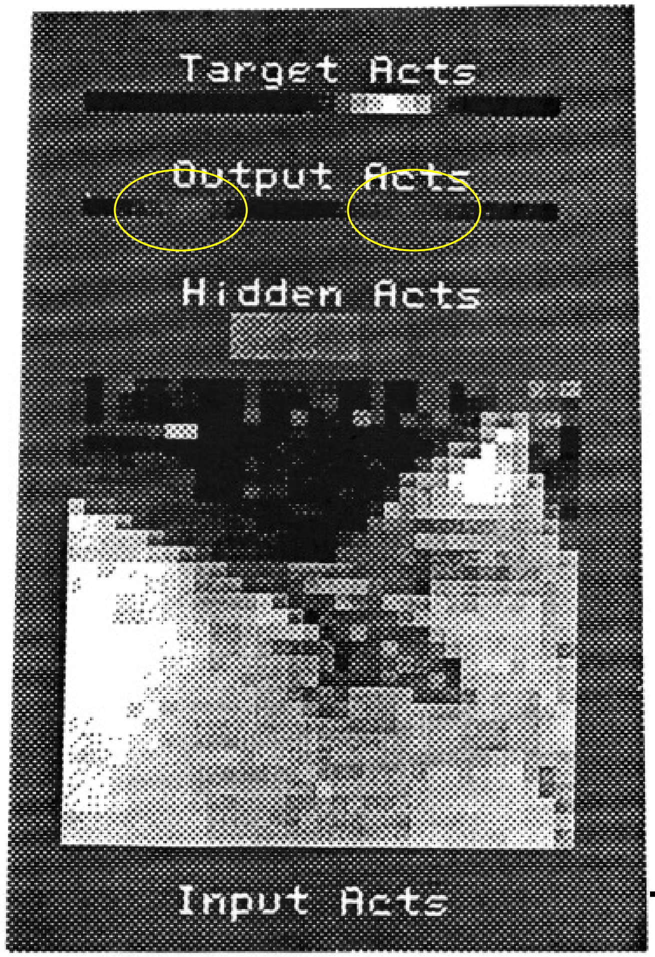
C = two land road



Fork Detection

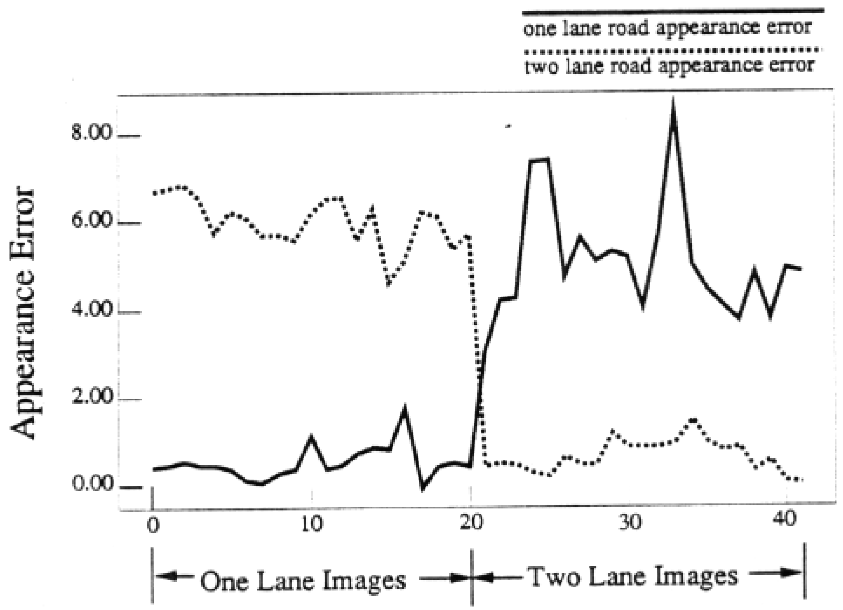
•A fork in the road causes a bimodal output pattern, which has high OARE.

•If the map shows we're approaching a fork, we can use OARE to detect when the fork has been reached.



Comparative OARE

- Comparing OARE for two networks can tell us when we have transitioned from one road type to another.

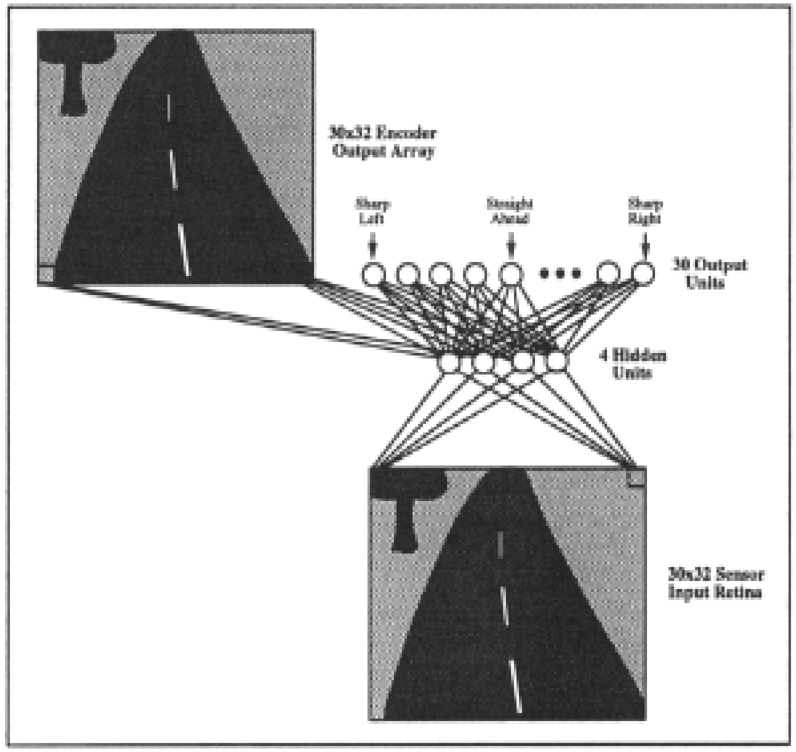


Input Reconstruction Reliability Estimation (IRRE)

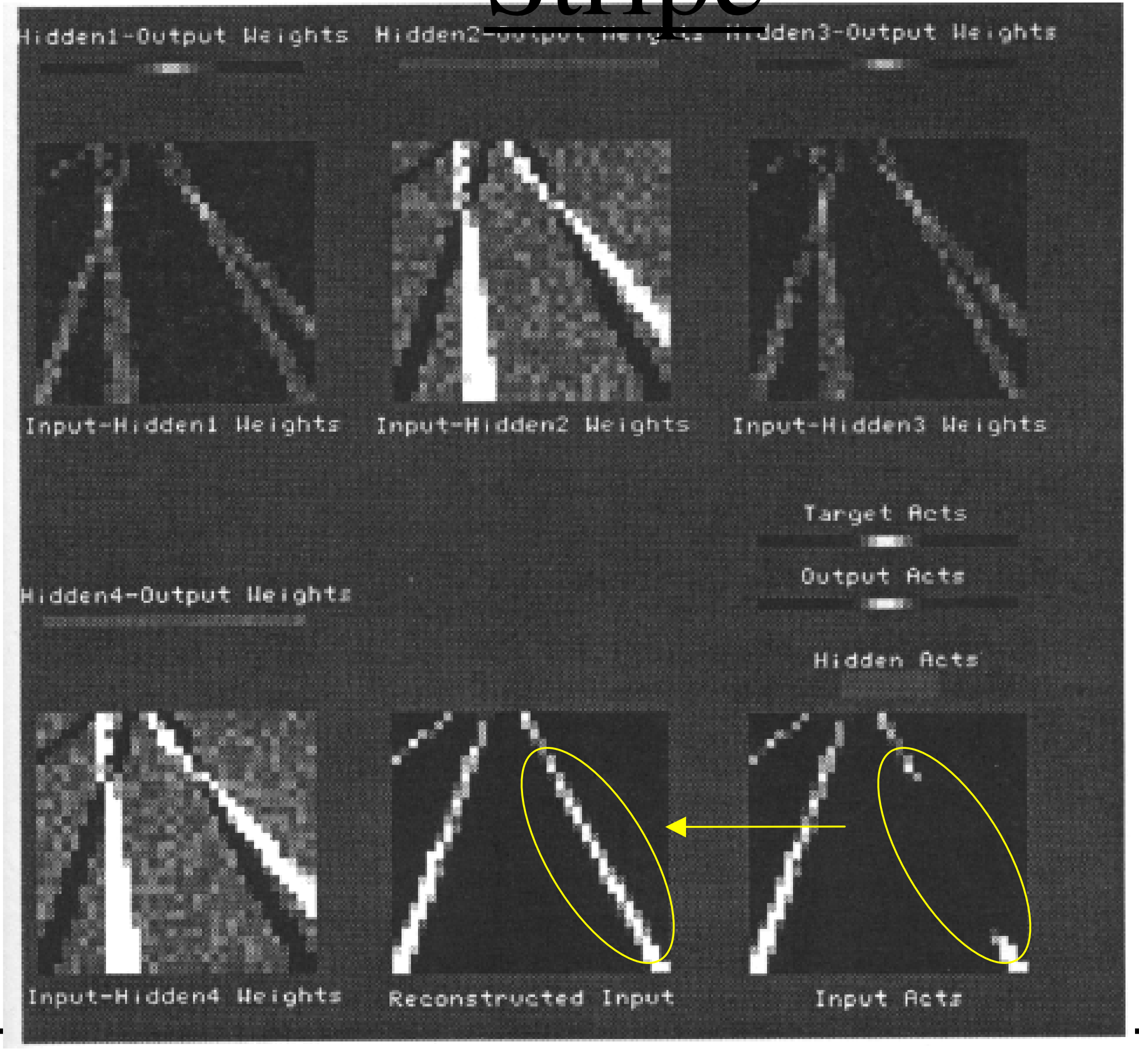
•Treat the network as an auto-encoder: require it to reconstruct the input.

•Hidden unit "bottleneck" extracts principal components of the image.

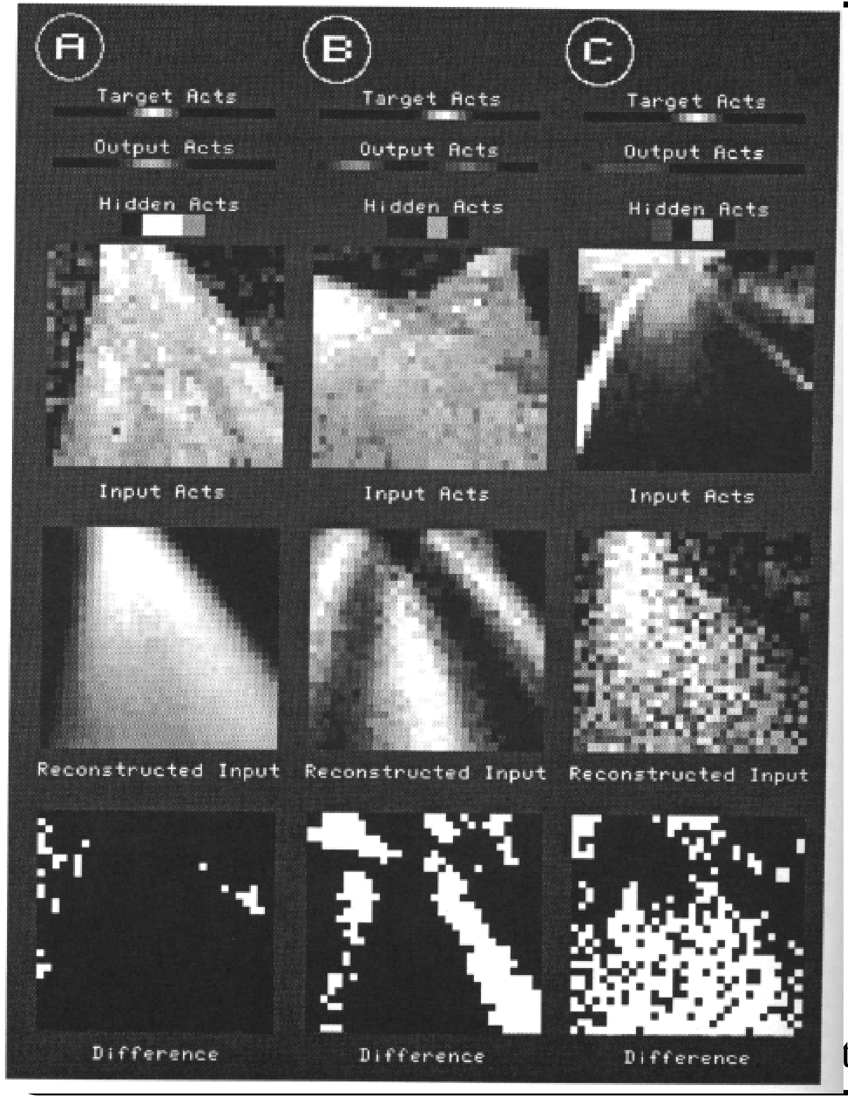
•Images far from the training set will not be reconstructed accurately.

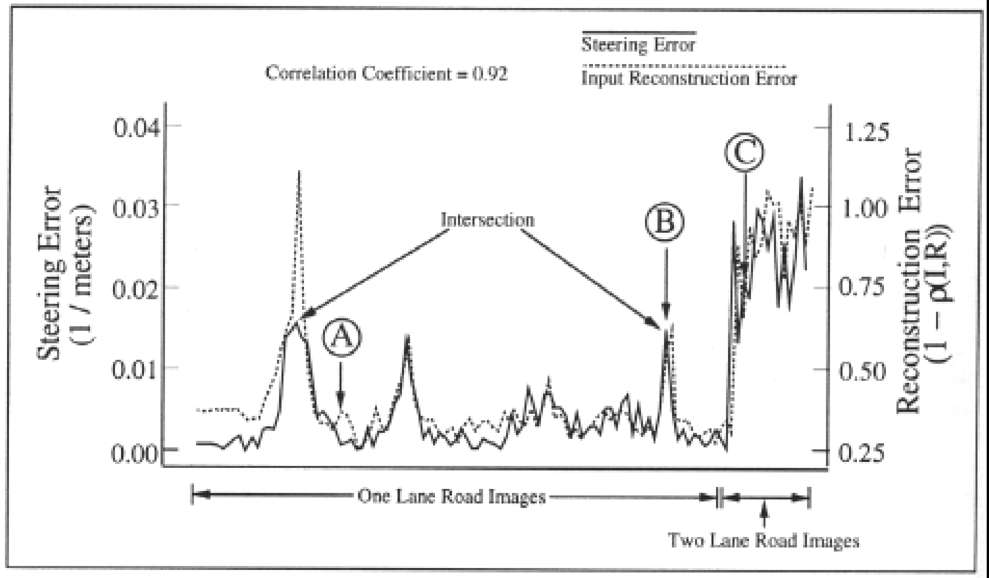


Auto-Encoder Fills Gap in Lane Stripe



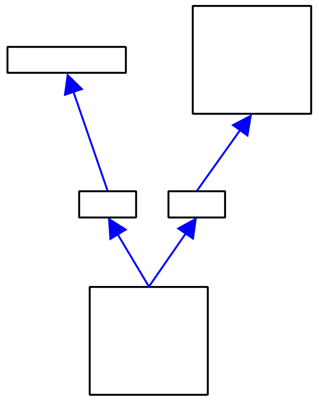
Difference In Reconstructed Image



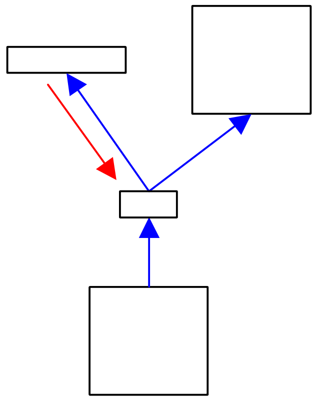


Drawbacks to IRRE

•Hidden units are forced to learn all the features of the input image, not just the ones relevant to the task. (Didn't seem to hurt ALVINN.) Solutions:



Separate set of hiddens for IRRE



Dont propagate error from IRRE outputs to hiddens

Speed Control

•If none of the networks appears to be reliable in the current situation, the system slows the vehicle and asks for further training.

•Other ways ALVINN controls speed:

• If networks are steering erratically, slow down.

• If networks are steering sharply, slow down.

Lessons Learned

1) Preprocessing Tradeoff: more preprocessing allows the net to solve harder problems, but it will be less flexible in new situations.

2) Importance of Modularity: by only requiring individual networks to handle relatively restricted situations, network training can be made fast and robust.

3) Viability of Hybrid Approach: to achieve high level behavior like route following, ANNs can (and currently must) be combined with symbolic

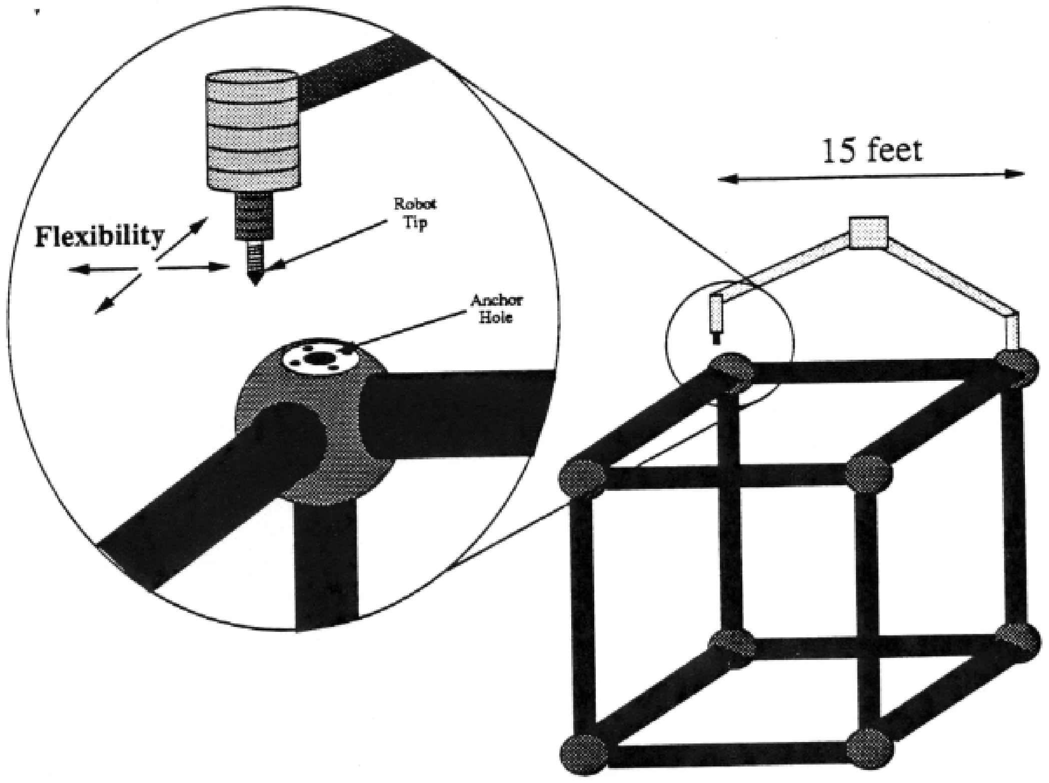
Lessons Learned

4) Analyzability: neural nets are not just black boxes. We can look at the internal representations and determine how they work.

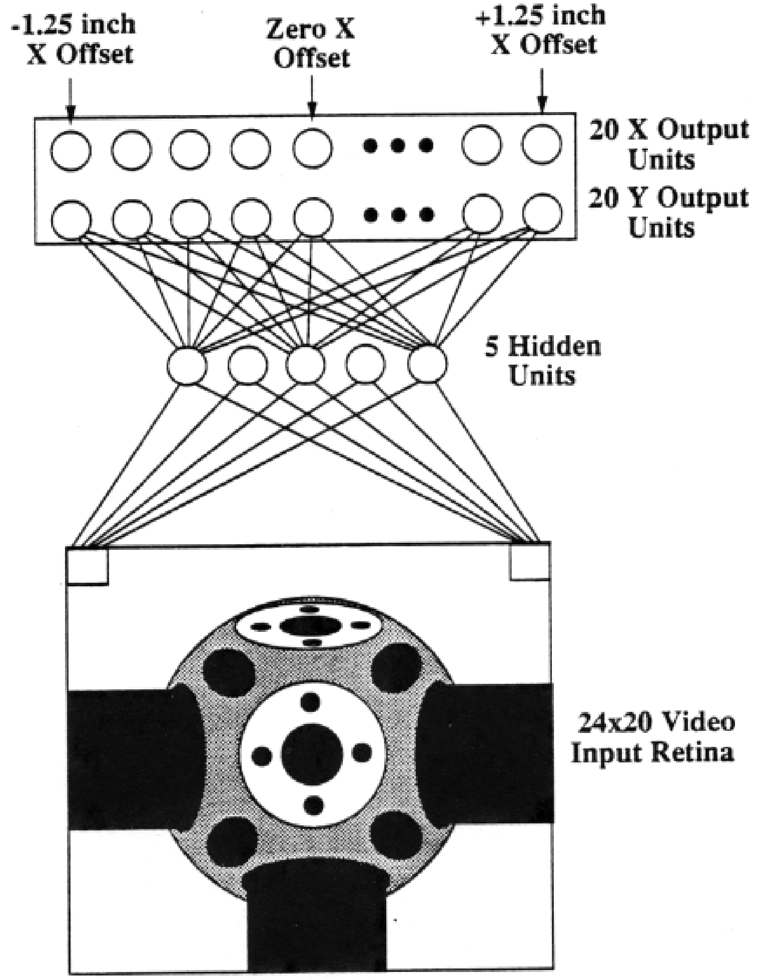
5) The techniques developed in ALVINN for robot driving are also applicable to other forms of vision-based robot guidance.

The simplicity of adapting ALVINN to new domains underscores the advantage that learning networks have over conventional hand-coded systems.

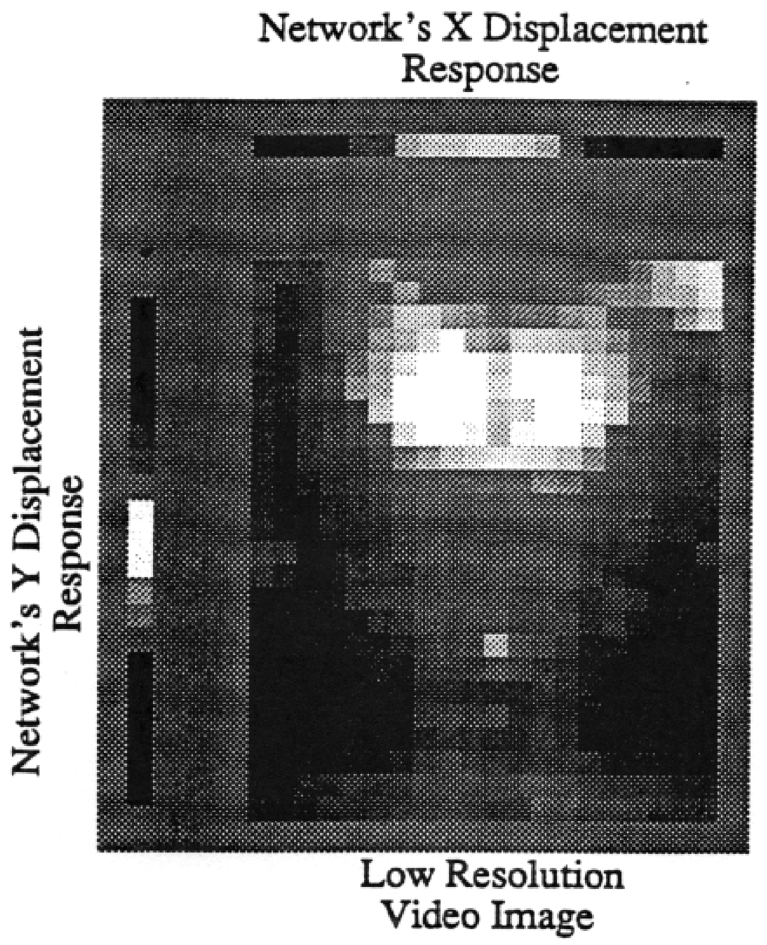
Self-Mobile Space Manipulator

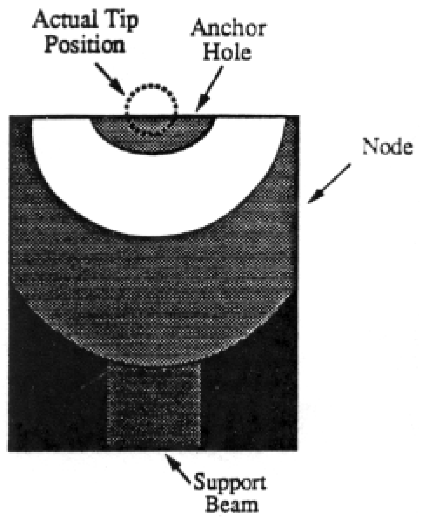


SMSM Network Architecture



What the Network Sees

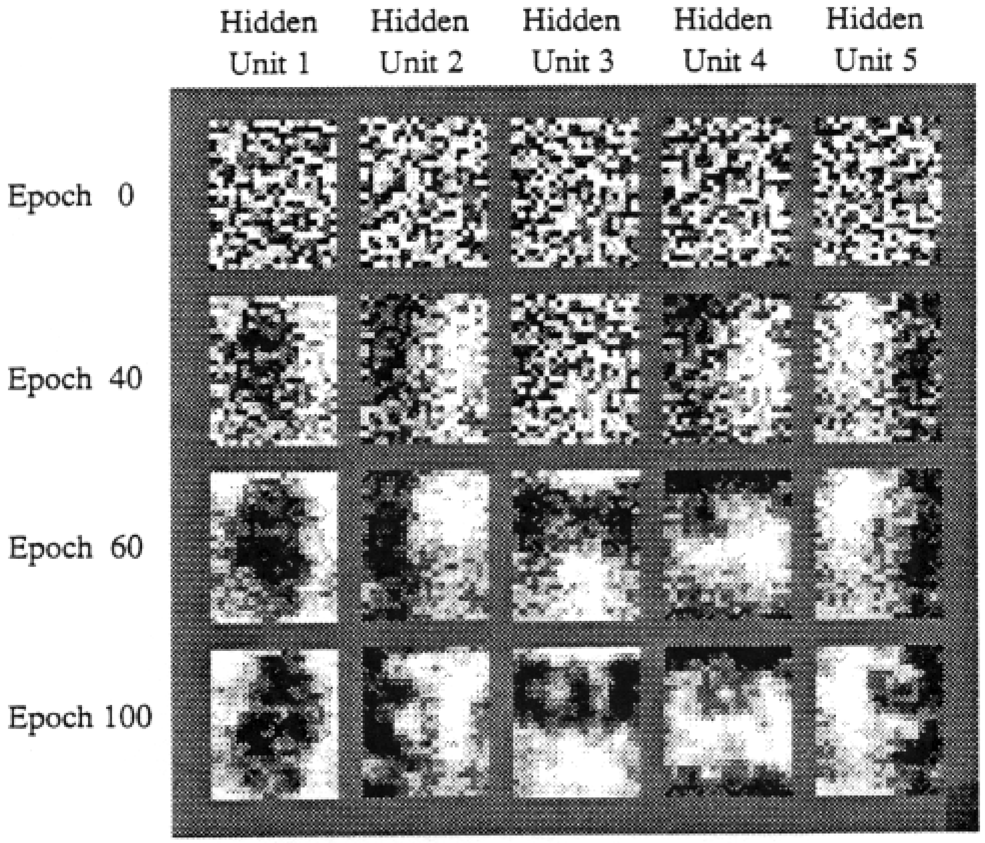

Network's X Displacement Response
Network's Y Displacement Response
Low Resolution Video Image



Learned Weights

•Weights start out random.

•With even a little training, a clear horizontal or vertical structure emerges in the input to hidden weights.



(movie)