Learning Perceptual Invariances‘ A Spatial Model

$$F = \log \frac{\bar{v}}{v}$$

where $\bar{v}$ is the long term variance of the unit, and v is the short term variance. By maximising F, we will jointly maximise the long term variance of the unit's output and minimise its short term variance. (The log function is used so that the derivative of F is easy to compute - see Appendix A.)

This learning method was used by Stone in a feedforward network presented with a sequence of one dimensional random dot stereograms. Each input was a pair of random dot stereograms, with the disparity between the two images varying in a sinusoidal fashion over time. The learning method was used to maximise F for the one output unit. After learning, the output of the network was highly correlated ($r > 0.$) with the disparity between the images.

The work presented here uses the same learning method to a similar problem in the spatial domain. Rather than having a sequence of images presented to the network over time, we have one large image where the disparity varies smoothly over the image. Additionally, instead of having one output unit, we now have an array of output units, and the learning rule is now applied to maximise F over all of the output units.

For brevity, the temporal model described by (Stone, 1) will be called the 'temporal model', whereas the model presented here will be called the 'spatial model'.

1.1 Related Work

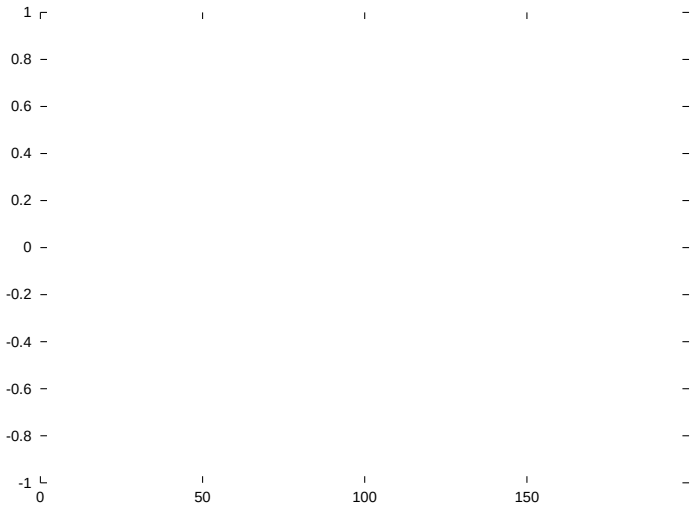
The spatial task used here is the same as the task used in (Beck

network. The output of the shared network was then copied into the appropriate part of the array of virtual output units.

For example, let the two input images be 00×1 pixels, and the input layer contain $\times 2$ input units. Both input images are therefore broken up into 100 patches of size $\times 1$. Patch n from both eyes is presented together as input to the network. The network computes the output of the network, and this output is then copied into unit n of the array of virtual output units. This procedure is repea

2.3 Learning Rule

The temporal model learning rule has been slightly adapted for the spatial model, although it essentially

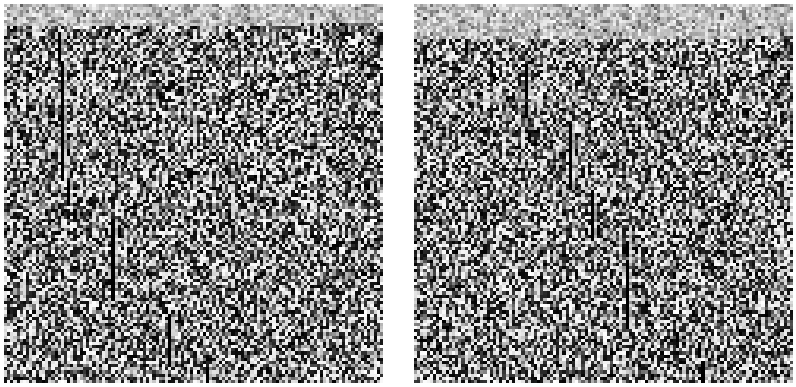

1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
50
100
150

3.2 Two Dimensional Network

A pair of random dot stereograms of size 120 × 120 pixels were created, such that the disparity between patches of the images was a gaussian function of the distance of the patch from the centre of the image (see Figure). These two images were then broken down into 5×5 non overlapping image patches. The shared network had 1 $(2 \times 5 \times 5)$ input units, hidden units, and 1 output unit, feeding into a virtual array of 1 00 (0 × 0) output units. Figure shows the network output after 1000 epochs, along with a plot of the merit function and correlation during learning. Figure shows how the outputs develop during learning. As can be seen from Figures and , the network has essentially learnt the disparity after 200 epochs, and the remaining 00 epochs are spent gradually improving the merit function.

(Note‘ The array of 0 × 0 disparity values and 0 × 0 output values are visualised as 0 × 0 greyscale images, with each disparity or output value represented by the pixel intensity‘ the larger the value, the brighter the pixel.)

3.2.1 Testing Network Performance



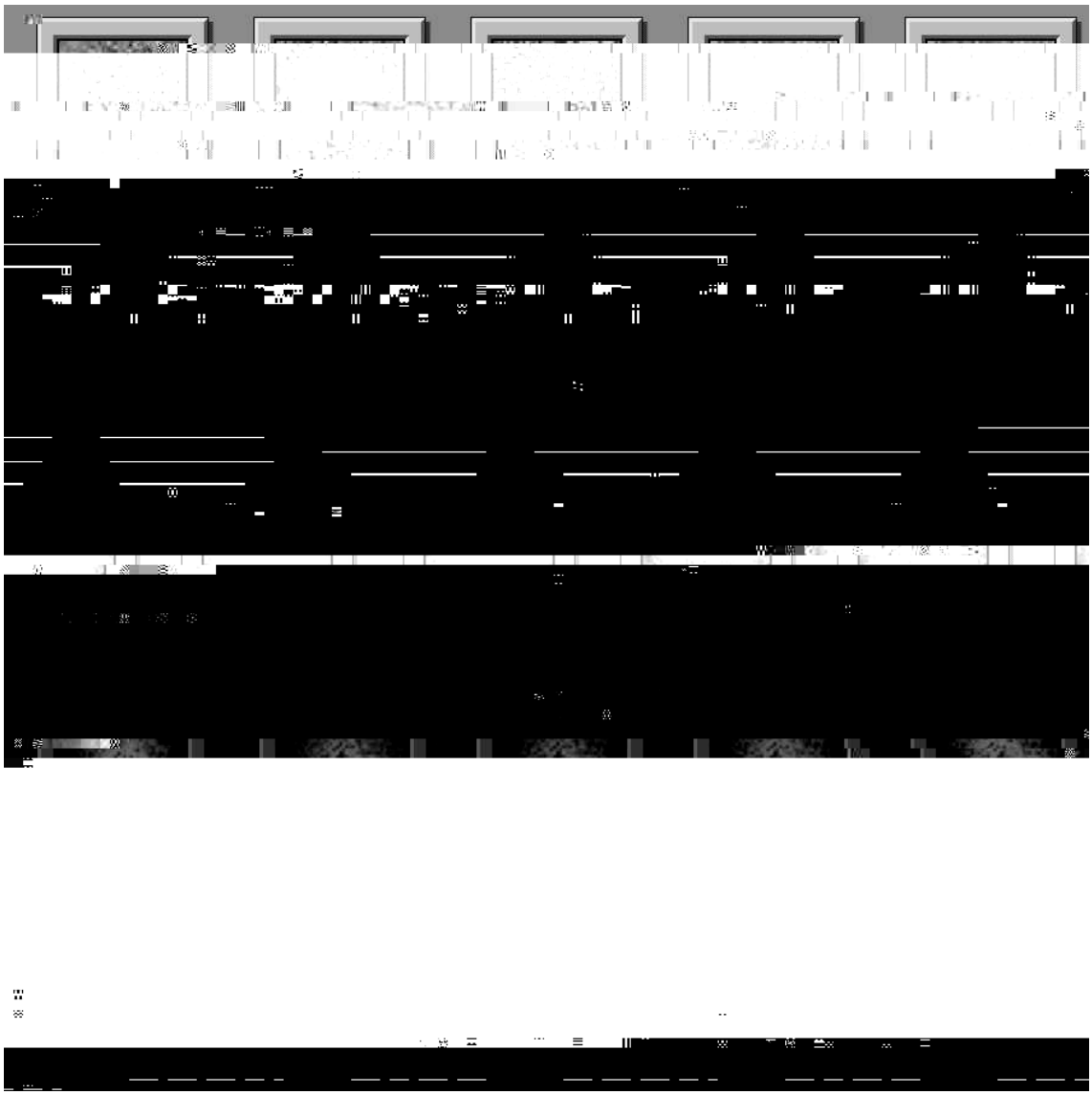


Figure ‘ Evolution of the outputs during learning for the 2D

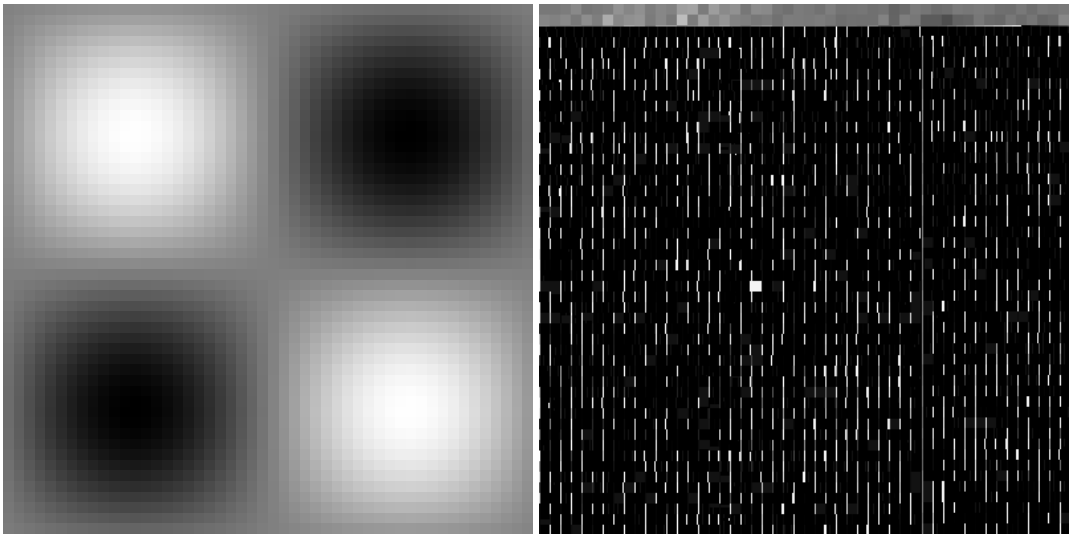


Figure 9: Results of testing the 2D network on a novel image pair. Left: Array of disparity values between the novel image pair. Right: Array of network outputs. The correlation between the disparity and the output is 0.90.

$$x = \sum$$

$$= x \qquad \text{(Identity transfer function)}$$

A.3 Calculating the Merit Function

The merit function, F, is defined as‘

$$F = \log \overline{\blacktriangledown}$$

A.4.1 Computing $\frac{\partial}{\partial x}$

The error δ

By symmetry, a similar expression can be derived for $\frac{\partial}{\partial x}$, so that the final δ is:

$$\delta = \frac{\partial F}{\partial x} = \frac{1}{} \frac{\partial}{\partial x} - \frac{1}{} \frac{\partial}{\partial x} \qquad (1)$$

A.4.2 Summary: Creating weight changes Δ

1. Calculate the output of virtual output units.
2. Calculate and using equations 1 and 2
3. Calculate δ for all output units using equations 1 and 1.
4. Back propagate errors to hidden layer units using equation .
5. Calculate weight changes Δ $= \delta$ for all weights.

Stone, J. V. (1[illegible]). A canonical microfunction for learning perceptual invariances. *P r pt on fort o n A so s C P*

Stone, J. V., & Bray, A. (1[illegible]). A learning rule for extracting spatio-temporal invariances. *or* , (*5*), 2[illegible] – *5* .

Williams, P. (1[illegible]1). A marquardt algorithm for choosing the step-size in backpropagation learning with conjugate gradients. Tech. rep. CSRP 22[illegible], COGS, University of Sussex.