Introduction to Neural Networks - Final Project

Detecting head orientation
Paul Fitzpatrick
paulfitz@ai.mit.edu

1 Introduction

The goal of this project is to train a neural network to classify the orientation of a centered image of a head as either left, right, up, or down.

Head pose gives information about what someone is paying attention to, and as such is important for social interaction and for mediating learning. Estimating head pose is an active area of research in computer vision. In child development, sensitivity to head and eye position matures through a reasonably clear series of milestones [5]. The earliest of these occurs within about six months, when an infant can decide whether someone is turned to gaze to their left or to their right, but cannot accurately determine their locus of attention. In this project, I attempt to replicate this level of competence with a simple neural network that can distinguish between faces turned left, right, up and down (see Figure 1).

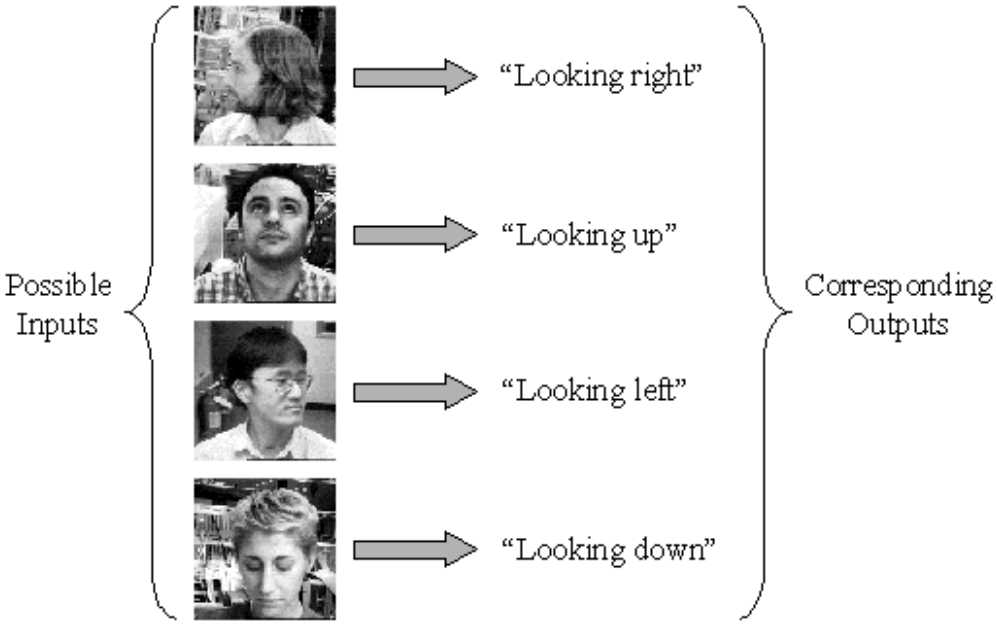

Figure 1: Desired performance of network. I have chosen to label directions throughout this paper to be from the point of view of the subject rather than the observer.

My approach is as follows :-

- ▷ First, a classifier is built which makes the same decision as the desired network, but does so using motion information – it needs to see the face turning from the frontal position to a new pose in order to classify that pose. This classifier is relatively straightforward to construct.
- ▷ The motion-based classifier is used to collect and label examples for training the neural network.
- ▷ When the neural network has been trained, it has the same functionality as the original classifier, except it does not rely on motion information. It doesn't have to see how the head moved to a given pose in order to classify that pose.

2 Joint attention

Why is gaze direction interesting? It is important for joint visual attention – "looking where someone else is looking". This is key to mediating social interaction and learning, currently topics of strong interest in humanoid robotics [4]. Mechanisms of joint attention include :-

- ▷ Mutual gaze – maintaining eye contact.
- ▷ Gaze following – locating the target of another's gaze.
- ▷ Imperative pointing – grasping towards an object that is out of reach. May be interpreted by parent of a child as a request for that object, and soon comes to have that meaning for the child too.
- ▷ Declarative pointing – drawing someone's attention to an object by pointing; not necessarily a request for the object.

I will focus on gaze following. Gaze following can in turn be broken down into a number of stages, as exhibited in child development :-

- ▷ The first stage is sensitivity to field (left or right). At 6 months, children show sensitivity to whether the care-giver is looking to their left or right (Figure 2).
- ▷ Then there is an "ecological stage" where the first object along the gaze direction as projected onto the child's point of view is chosen.
- ▷ This is followed by a "geometric stage" where full three-dimensional gaze information is used, but the child will not turn to search behind it even when that is indicated by the geometry.
- ▷ Finally there is a "representational stage" where the child will for the first time turn to look behind her if necessary.

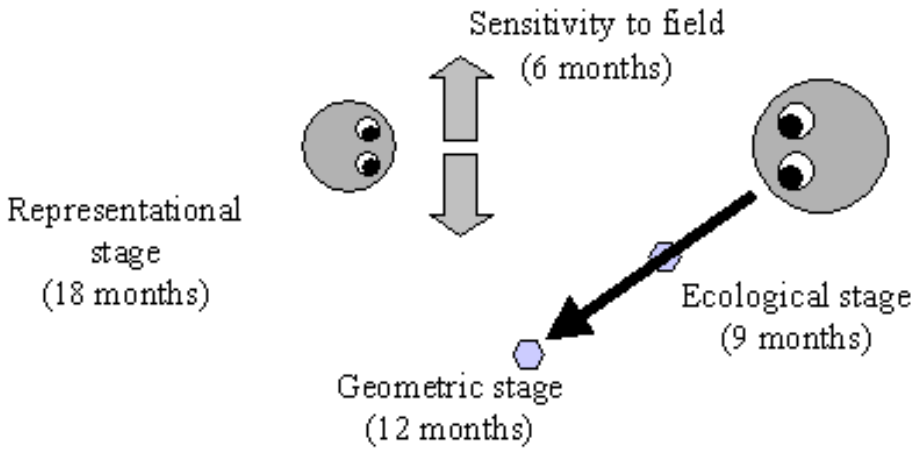

Figure 2: Stages of gaze following in child development.

Again I will narrow my focus, to the first of these stages – sensitivity to field. I will extend this to classifying head direction as either left, right, up, or down. I will cast the problem as pose classification, rather than pose estimation, but will return to the problem of pose estimation towards the end of the paper.

Also, in the above, "gaze direction" is used loosely to apply to both head pose and eye orientation. I will only treat the head pose aspect.

3 Collecting training data

Training data was acquired as follows :-

- ▷ A subject sat at a fixed distance from a camera.
- ▷ The subject faced the camera frontally.
- ▷ The subject turned his/her head either left, right, up, or down.
- ▷ An arbitrary feature on the central part of the subject's face was tracked.
- ▷ The direction of motion of this feature relative to the motion of the overall head was used to classify the turn (see Figure 3).
- ▷ An image of the head in its final position was taken and labeled with the appropriate classification from motion tracking.

Figure 3: This figure illustrates the activity of the motion-based head direction classifier. Upper row: Subject presents face frontally, then turns to subject's right. Bottom row: Tracker follows feature of face that is initially centered (shown by the series of circles). This feature moves rapidly to the left in the image, relative to the movement of the face itself (shown by the series of ovals). Hence the motion is classified as a right turn, and the final image in the series is recorded as a training example of a face turned to the right.

The neck has three degrees of freedom: pitch, yaw, and roll. It would be nice to estimate all three; however, I only consider two of them, pitch and yaw. When a person is facing the camera and makes a neck rotation in pitch or yaw, the outline of the head moves less than individual features on the head (see Figure 4). This is not necessarily true for roll, which makes it harder to deal with using this simple type of motion-based classifier.

I will now give details of how I tracked features on the head and the overall motion of the head itself.

3.1 Tracking head features

The motion-based head orientation classifier was created by modifying an object tracker, designed to keep the camera pointing at a moving target. For orientation detection, the camera was held steady and the output of the tracker was instead used to indicate the direction of the head turn.

The tracker receives input from the camera at 30 frames a second, and estimates the motion of a small patch in the center of the image. The standard, simplest tracking technique is to choose a small rectangular patch at the center of the image as the "template", and then to look for the patch in the next frame that is the best match for the template. The goodness of a match is measured using the Euclidean distance between the patches, or other simple

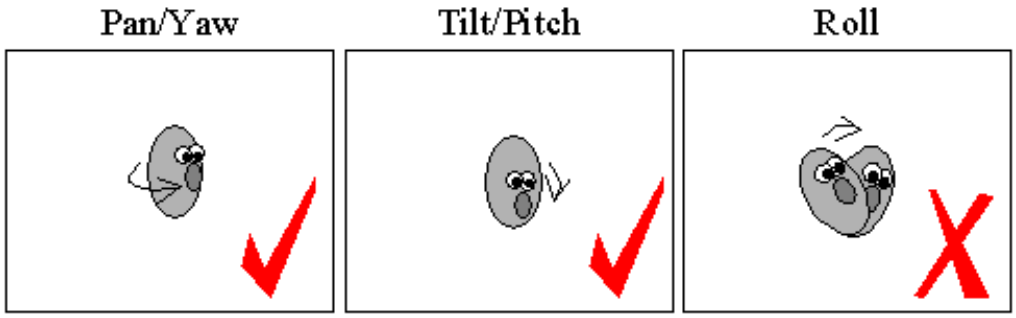

Figure 4: This figure shows why it is harder to apply motion-based head orientation detection to head rolls, compared to head movements in yaw ("no" nod direction) and pitch ("yes" nod direction). For yaw and pitch, the rotation of the head is such that the projection of the overall face motion is less than individual feature motion, and the method works. For roll, the axis of rotation of the head is such that the overall motion of the head is comparable to individual feature motion, and the method fails.

measures. One problem with this type of tracking is how to maintain the template correctly. Clearly the target may change appearance somewhat as it moves, particularly during rotation, and the template should change to reflect that. One possibility is to replace the template at each iteration with the best matching patch. This can cause "template drift", where the template gradually slips away from the target until the tracker is following something else entirely. Another possibility is to mix the two with some weighting. This reduces template drift, but does not eliminate it.

For my purposes, template drift is not a particularly important issue. I am trying to get an approximate measure of the direction of feature movement; for this, it does not matter if I drift from tracking one feature to tracking a nearby one on the face. So building a motion-based head orientation classifier from a tracker was quite straightforward, and the end-product was robust.

3.2 Tracking the head

To distinguish translational head movements from the rotations of interest for this project, it is necessary to track the overall position of the head so that the relative motion of features on the face can be compared to the motion of the head itself. I investigated three alternatives for doing this :-

- ▷ Image differencing. Successive frames can be subtracted from each other, then thresholded to find areas of motion.
- ▷ Face detection. A face detector has been implemented for Cog, the vision platform I was working on. Unfortunately it is not designed to work with non-frontal views of a face [11].
- ▷ Tracking ovals [1]. The outline of the moving face can be approximated as a deforming oval for tracking.

None of these worked robustly enough for the patience of my subjects. So instead I made the system less autonomous by constraining the motions the subjects could make in front of it. Translational motions were therefore simply mis-classified as some form of rotation and manually eliminated from the training data if they occurred.

4 Faces collected

Approximately 270 snapshots of faces were collected. The bulk of the faces were of one subject, 'Paul' (see Table 1). Smaller numbers of images were collected from four other subjects. Samples of the faces are shown in Figure 5.

Subject	Number of batches	Total number of images
Paul	3	170
Artur	2	40
Jessica	1	22
Hideki	1	20
Ulysses	1	20

Table 1: Number of examples collected from various subjects.

Figure 5: Examples of the faces collected.

Figure 6: Variation in head pose of a single subject.

- ▷ Some care was taken that the faces were on roughly the same scale, and approximately centered in the image. This was not done rigorously, since automated techniques for doing this would be noisy. If the system relied on extreme precision, it would not be possible to make it autonomous, which is a long-term goal.
- ▷ The background was changed between batches by pointing the camera in different directions. Backgrounds vary from a bright white computer monitor to a dark metallic backplane.
- ▷ Lighting conditions were not explicitly altered, but did vary to some degree.

The subjects were directed to vary the extent of their head turns on their own whim. Figure 6 shows examples of the variation across the images collected for a single individual.

5 Network architecture

I chose a very simple network topology to begin with (see Figure 7). The raw images were fed almost directly to the input layer of a neural network. This projected to a single hidden layer with a small number of neurons. This in turn projected to the output layer, which had a neuron for each of the four possible orientation classifications.

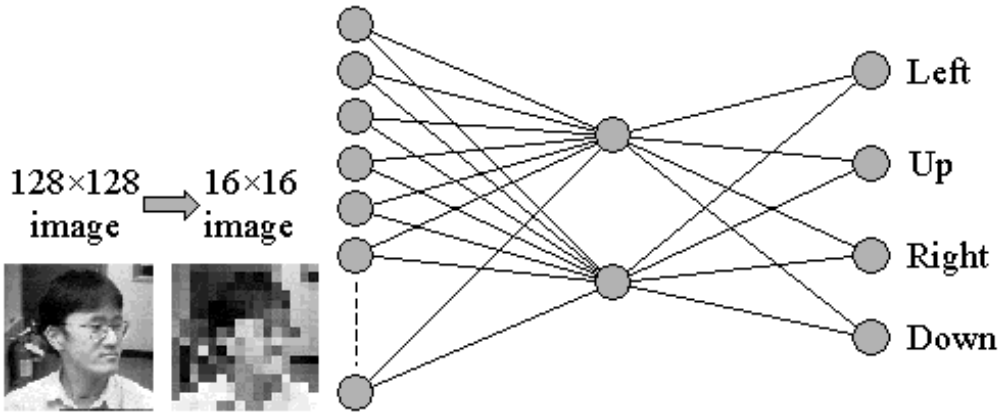

Figure 7: Network architecture. The image resolution is first reduced from 128×128 to 16×16, and then the image is fed to the input layer of the network. This projects onto the hidden layer, which has only two neurons in this example. These project onto four output neurons.

Initially I chose to have just two hidden neurons. This was to force the network to make the four-way choice of up, down, left, or right through a two-dimensional representation. I hoped that even though the network was being trained as a classifier, this intermediate representation might correlate with the magnitude of the yaw/pitch angles. Of course, the hidden layer neurons are just perceptron-like combinations of the inputs, so this representation is being generated without the power of the full neural network. I return to this idea towards the end of the paper.

6 Results of backpropagation

I trained the neural network to perform the head orientation detection task using backpropagation. The desired output of the network was to have a "1" at the output neuron corresponding to the direction of gaze, and "0" at the other neurons (akin to the hand-written digit classification example). I restricted myself to training for discrete decisions like this, rather than encoding degrees of rotation, because the output from the motion-based classifier was too noisy to give accurate training data for this more nuanced task.

Figure 8 shows the behavior of the error during a sample run of backpropagation on the network.

- ▷ The "test data" in this case was every second image. Since this means that subjects in the test set are also represented in the training set, the classification error on the test set is misleadingly low (about 2%, meaning just a few mistakes over the entire test set). I will give more meaningful measurements in the next section.
- ▷ The classification error on the training set quickly falls to zero. But even after this, the error on the test set continues to fall as the squared error on the training set is reduced.

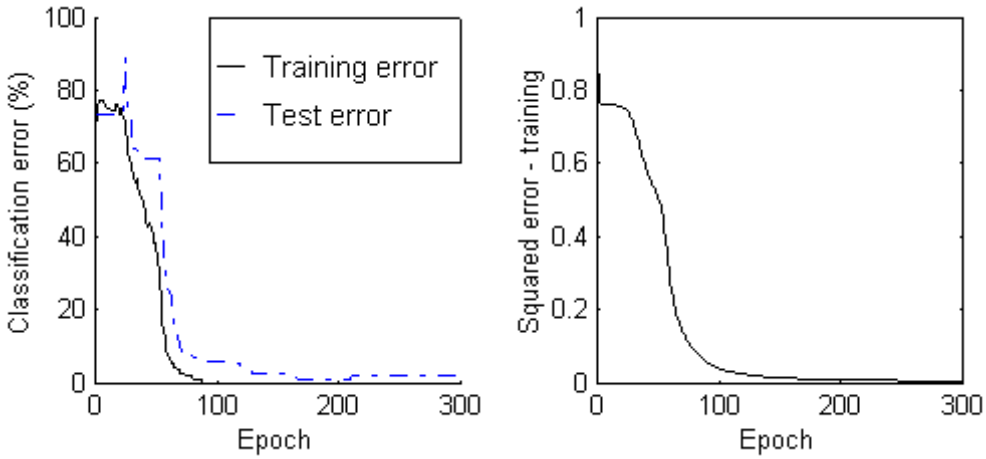

Figure 8: Training run.

The weights projecting onto each of the two hidden units from the inputs are easy to understand. They typically show a central circular patch, with excitation on one side and inhibition on the other (see Figure 9). The patches for the two neurons are rotated by 90° to each other. The weights and biases for the output neurons are shown in Table 2.

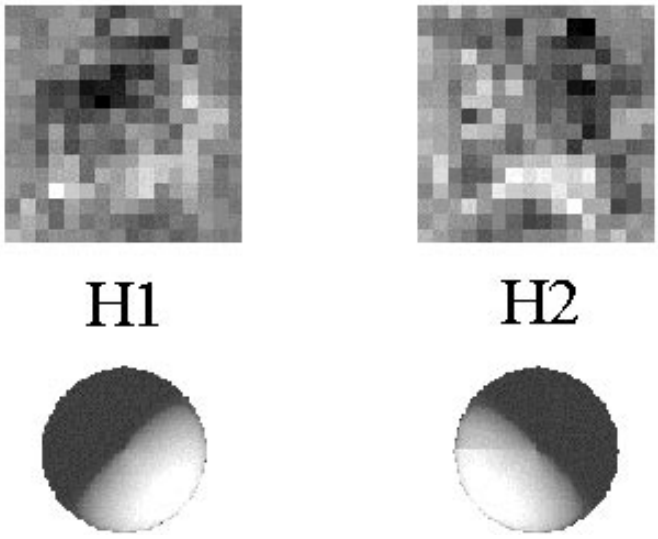

Figure 9: Upper row: the weights from the input image to each of the two neurons in the hidden layer (labeled H1 and H2 for reference). Lower row: what these weights look like if you squint a little.

Output neuron	**Weight from H1**	**Weight from H2**	**Bias**
"Right"	-8.4	8	-3.7
"Up"	-8.2	-8.3	4.1
"Left"	8.0	-8.3	-3.7
"Down"	7.4	7.6	-11

Table 2: Weights and biases for output neurons.

▷ The pattern of the weights shows a bilateral symmetry around the vertical axis of the image, in the following

sense. The weights to the hidden layer neurons are approximately mirror images of each other, and the weights projecting from these onto the *Left* and *Right* outputs are almost identical, with a sign change.

▷ The bilateral symmetry inherent in the classification problem could have been used to constrain the network, but it is satisfying that the network determined the symmetry itself.

Output neuron	**Weight from H1**	**Weight from H2**	**Bias**
"Right"	-	+	$\frac{\theta_{low}+\theta_{high}}{2}$
"Up"	-	-	θ_{high}
"Left"	+	-	$\frac{\theta_{low}+\theta_{high}}{2}$
"Down"	+	+	θ_{low}

Table 3: Cartoon version of Table 2, showing the patterns within it.

7 Classification results

Figures 10, 11, and 12 show results for the performance of the network on subjects not represented in the training set.

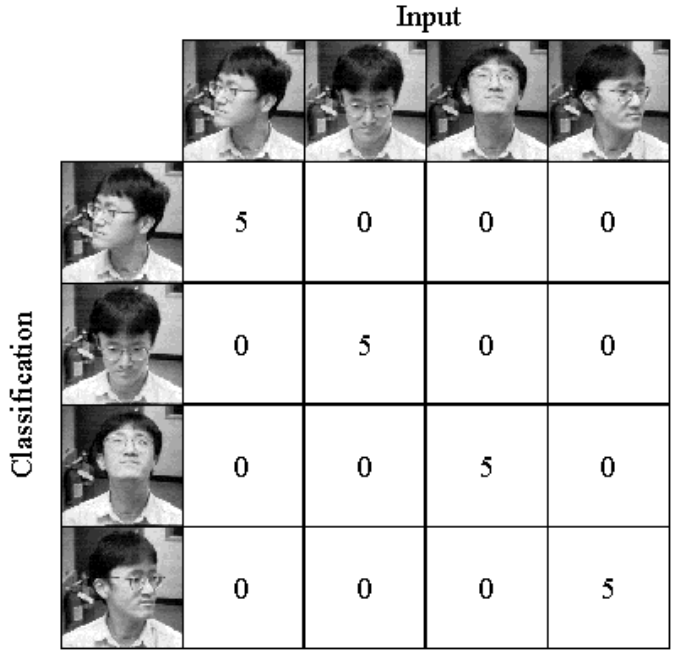

Figure 10: Confusion matrix for subject 'Hideki' when the network was trained on subjects 'Paul', 'Artur', and 'Jessica'. This subject wore glasses, unlike any of the training subjects. The network correctly classified the 20 head pose examples taken from this subject. The images in this figure and the figures to follow are representative examples of the test poses presented to the network, and are used as icons for their respective orientations.

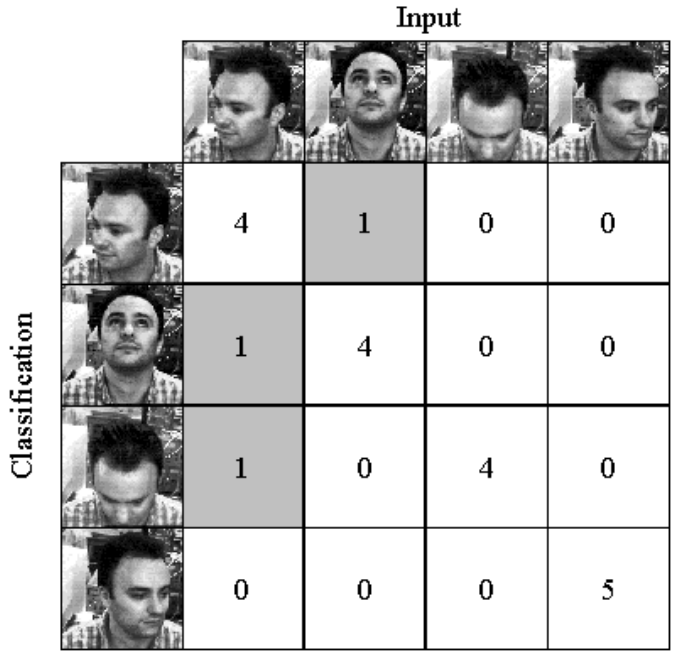

Figure 11: Results for subject 'Ulysses'. Network trained on same subjects as for Figure 10. The network makes 3 classification errors on a total of 20 inputs.

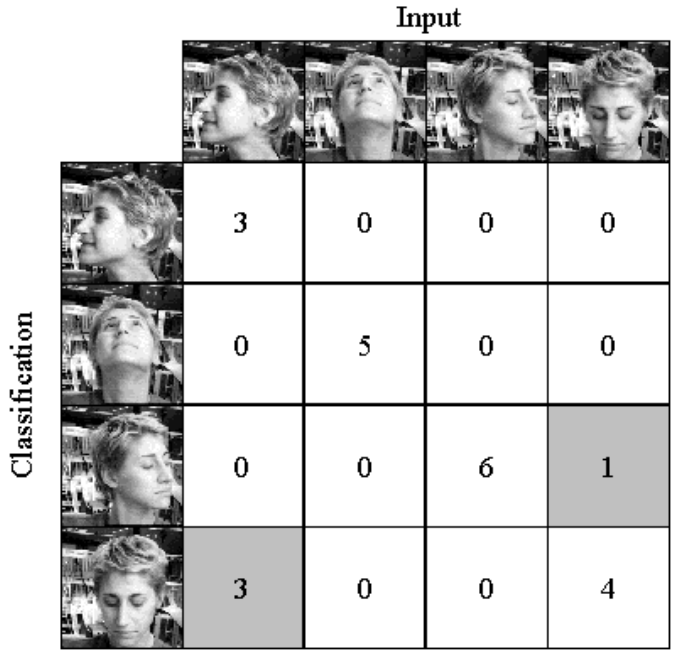

Figure 12: Results for subject 'Jessica'. Network trained on 'Paul' and 'Artur' only. The network makes 4 classification errors on a total of 22 inputs. It confuses *Right* with *Down* quite badly. Note that this subject has fair hair, unlike subjects 'Paul' or 'Artur'.

8 Extracting a coordinate system from the network

Before the non-linearity in the two hidden layer neurons, weighted sums are taken over the input image. Since, for a given subject, this image will vary continuously with head pose, the sums will also vary continuously. Since these sums at the two hidden layer neurons completely determine the activity of the output nodes, there must be enough information in these two numbers to distinguish left, right, up, and down. These two statements taken together suggest that the two sums might give a coordinate system for representing head pose.

Table 2 suggests that the sum of the two neurons may correspond to the vertical direction, and the difference of the two may correspond to the horizontal direction (the table shows weightings of the sums *after* the non-linearity, but this doesn't affect the argument). I have plotted these values in Figure 13 for two head movements, one in the "yes" direction, and one in the "no" direction. The lines form an approximate cross-shape, which is encouraging. Figure 13 shows results for four series of head movements, all in the left-to-right direction, with different pitches. The lines are non-intersecting, although by no means grid-like. Hence they do form an implicit map of head pose.

The sums I'm plotting are of course formed in an extremely simple way, so I don't mean to suggest that there is anything particularly mysterious or profound about them. Their biggest drawback is that they are subject-dependent, such that the map shifts around somewhat from individual to individual.

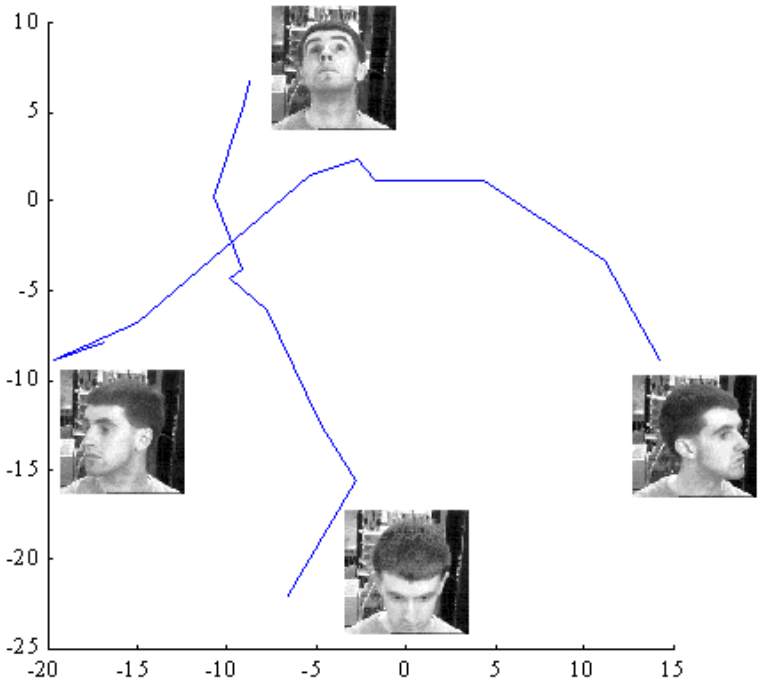

Figure 13: Traces for two head movements. One was a movement in yaw (traced as the curved, roughly horizontal line), and one was a movement in pitch (traced as the roughly vertical line).

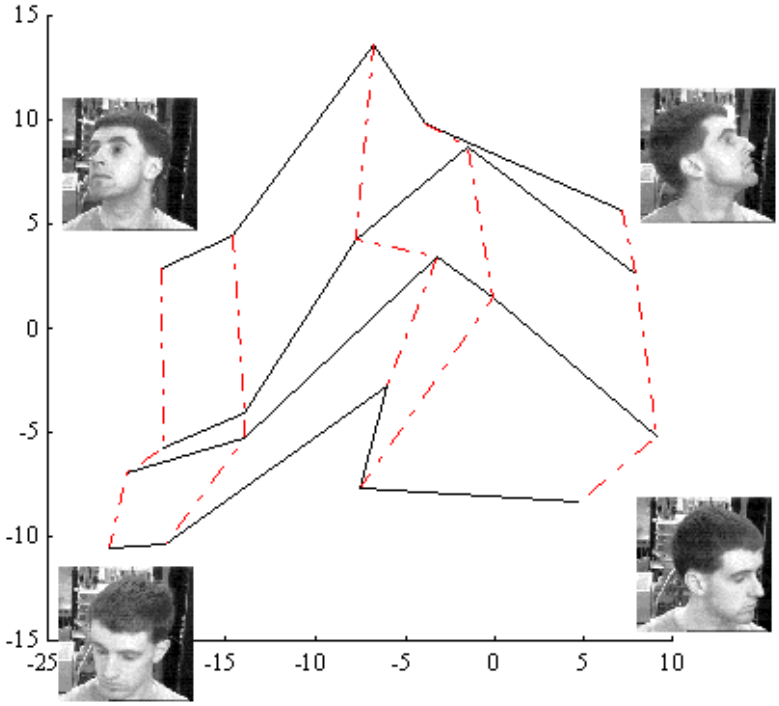

Figure 14: Traces for a series of head movements. The movements were in pitch (solid lines). The dotted lines join points of approximately equal yaw.

9 Conclusions

I have presented a very simple network architecture in this paper. I did try many other architectures, and various forms of feature selection on the input. None gave performance improvement worth speaking about, and the two-hidden-neuron architecture was so much easier to understand and explain that I have chosen to present it instead of the more complex models. One refinement that did prove worthwhile was to add a circular mask to the image to remove some of the background, and to perform a normalization for lighting correction.

The trained classifier extracted very simple features from faces to classify them. It uses just the gross patterns of light and dark across the head, relying on the heuristic that hair is often darker than skin. This means the resulting classifier is both racist and "blondist". More generally, the classifier suffers from the problem that, while the patterns of light and dark it measures do seem to give a good characterization of head pose, the mapping from those measures into angles varies from individual to individual. To tune the network to a new individual, the weights from the input layer to the hidden layer need not be changed, but the weights and more importantly the biases for the output layer may need to be altered somewhat.

Returning to the child development theme, this problem does not seem so crucial, since it is only necessary for the classifier to work with a small number of people – the child's parents. Perhaps the neural network classifier could act as an intermediate stage to train a more general pose estimator, just as the motion-based classifier was used to train the neural network itself.

References

[1] Artur M. Arsenio and Jessica L. Banks. How to find friends and keep them: people detection and tracking by a humanoid robot, unpublished. 1999.

[2] S. Baron-Cohen and H. Ring. A model of the mindreading system: neuropsychological and neurobiological perspectives. In P. Mitchell and C. Lewis, editors, *Origins of an understanding of mind*. Lawrence Erlbaum Associates, 1994.

[3] D. J. Beymer. Face recognition under varying pose. In *Computer Vision and Pattern Recognition*, pages 756–761, June 1994.

[4] R. A. Brooks, C. Breazeal, R. Irie, C. C. Kemp, M. Marjanovic, B. Scassellati, and M. Williamson. Alternate essences of intelligence. *AAAI*, 1998.

[5] G. Butterworth. The ontogeny and phylogeny of joint visual attention. In A. Whiten, editor, *Natural Theories of Mind*. Blackwell, 1991.

[6] C. Brazeal (Ferrell) and B. Scassellati. Infant-like social interactions between a robot and a human caretaker. *Special issue of Adaptive Behavior on Simulation Models of Social Agents*, 1998.

[7] C. Brazeal (Ferrell) and B. Scassellati. A context-dependent attention system for a social robot. *IJCAI*, 1999.

[8] A. H. Gee and R. Cipolla. Estimating gaze from a single view of a face. In *12th International Conference on Pattern Recognition*, volume 1, pages 758–760, Jerusalem, Israel, October 1994.

[9] T. Horprasert, Y. Yacoob, and L. S. Davis. Computing 3-d head orientation from a monocular image sequence. In D. H. Schaefer and E. F. Williams, editors, *In Proceedings of SPIE*, volume 2962, pages 244–252. SPIE Press, 1997.

[10] D. D. Lee and H. S. Seung. A neural network based head tracking system. *Advances in Neural Information Processing Systems*, 10:908–14, 1998.

[11] B. Scassellati. Eye finding via face detection for a foveated, active vision system. *AAAI*, 1998.

[12] B. Scassellati. Imitation and mechanisms of joint attention: A developmental structure for building social skills on a humanoid robot. *To appear as part of a Springer-Verlag series*, 1999.

[13] I. Shimizu, Z. Zhang, S. Akamatsu, and K. Deguchi. Head pose determination from one image using a generic model. In *4th International Conference on Automatic Face and Gesture Recognition*, pages 100–105, April 1998.

[14] R. Stiefelhagen, M. Finke, J. Yang, and A. Waibel. From gaze to focus of attention. In *Third International Conference on Visual Information Systems*.