# *Contents*

# *Head Pose Estimation Method*

## *Introduction*

Nao serves as a link between the smart-home environment and the user. Elderly are less likely to accept new technology in their home. By using Nao, users are able to profit from a smart-home environment without the need to interact with the system using difficult immobile interfaces. Nao informs the user and controls the environment without the user to read any manuals or to learn how to use the system. To let elderly accept Nao into their homes it is important to improve the interaction and make it as natural as possible.

The user should to be focused on Nao, when Nao is informing the user about important information concerning his/her health. Nao should therefore check during interaction if the user is still engaged in the interaction, and not, for example, looking at the television. For this end one needs to infer the gaze direction of the user.

Normally the gaze direction can be estimated by determining both the head and eye orientation (Wollaston, 1824). But Nao's cameras do, most of the time, not allow for the determination of eye orientation. The resolution is simply not high enough. But one cue, namely head orientation, might be enough to give an estimate of the gaze direction. In addition, a measure of head orientation might also allow Nao to use conversational etiquettes like joint attention and reciprocal gaze aversion (Kozima, Nakagawa, & Yano, 2003).

## *Neural network approach*

To determine the head orientation one needs to know the head pose. Normally head pose is expressed in a pitch, yaw and roll angle (see

Figure 1). There are different approached to determining this from an image or video-stream. Some solutions use feature tracking to determine head pose. Features within the facial region, e.g. eye corner, mouth corner, nose tip, are detected and tracked for every frame. The location of these features and the distance between them determine the head rotation. This method works rather accurate but there are some downfalls. Firstly, the detection of these features using templates will only work if the resolution of the image is high enough. Secondly, for more extreme rotations of the head some features will inevitably be obstructed. Other options use a three dimensional model of the head to determine the rotation of the head. This method also used features. The problem is that for every face a three dimensional model should be created.

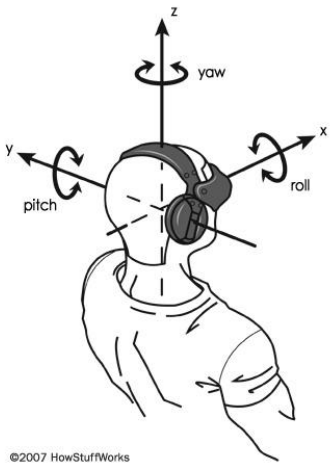


**Figure 1. Pitch, yaw and roll.**

Our approach is based on the neural network approach as proposed by Voit, Nickel and Stiefelhagen (2005). For their neural network approach, they used the training set "Face Pointing04" from the PRIMA Team in INRIA Rhone-Alpes. Their network is able to predict the head orientation with a mean error of about 12.5° for both the yaw and pitch.

We adapted their approach because the results were not satisfying. The problem with their measures is that they tested their network with a test-set coming from the same database as their

training set, which normally would lead to better results for that specific contextual set. We found that when their network is presented with images from a camera the results are highly dependent on the lighting conditions.

## *Our method*

Before we can start to train a neural network it is important to create a training, validation and test set. The idea is to filter the original image coming from Nao's camera and only keep the information important to determine the head orientation.

## *Creating training set*

In addition to the "Face Pointing04" database (see Figure 2) we also used Yale's Face pointing database. Yale's face pointing database has faces that are rotated with smaller angles, and it's images are taken with different lighting conditions. Using both of the databases allows us to train a more general neural network.

**Figure 2. Image from the Face Pointin04 database.**

The first step is to filter out all the context information, leaving only the face image. We used OpenCV to detect the face in a picture using their ready available object detection algorithm (Viola & Jones, 2001) and selected only the part of the information that is not influenced by hairdo, and background. This resulted in an image with a 4:9 ratio (see Figure 3).

**Figure 3. 4:9 selection of the facial region.**

The important information in this picture for head position estimation are, according to Voit et al. (2005), the edges in the picture together with a normalized gray-scaled image of the same region. We found that the gray-scaled image does not add extra information for head pose estimation, and therefore only saved the edge detected image. In order to retrieve the edges, the color image is transformed to a gray-scaled image and the values are normalized. We used a laplacian operator to retrieve the edges. The image is scaled to a resolution of 40x90 pixels (see Figure 4). The process is automated and creates a total of 6330 images.

**Figure 4. Scaled down Laplacian.**

## *Training the neural networks*

We used Matlab's neural network toolbox to train the network with the dataset we created. We created 10 different sets of neural networks. For each set the neural network size and the training data is varied. Due to memory limitations, we were not able to train one neural network with all data. We resorted in training multiple networks with different subsets of the dataset. Some network sets were trained with only the Face Pointing04 data while others were trained with only Yale data or a combination of both datasets. As a result, the different network sets had different properties.

Each network set is divided in two networks; one is used to determine the pitch, the other is used to determine the yaw. Both networks are trained with the exact same trainings data, but with different target data. The target for the pitch network would be the pitch angle of the depicted face in degrees. The target for the yaw network would be the yaw angle of the depicted face in degrees.

## *Employing the neural networks*

To use the trained Matlab neural network structures we created a c++ program that is able to communicate with the Matlab console. A webcam image is captured and transformed in a similar fashion as done for the creation of the training, test and validation set. But in this case, the search

area is limited by the previous frame (t-1), and the minimal frame used for the object detection algorithm is limited by a 40x40 frame size. This makes the rather slow face detection procedure faster, resulting in a higher frame rate.

We used all ten neural networks do determine the head position. Each set has its own properties; some neural networks are more precise in determining the head position given ideal lighting conditions while others are less accurate in determining the head position but are able to do this for a large range of lighting conditions. For now, the results for all these networks are combined to estimate the head position. The average of all 10 pitch or pan angles is taken. To make the program more robust to peaks in the neural network output, we implemented a low pass filter to give more steady results.

Figure 5 shows the result of all networks in graphs two to eleven. The first graph shows the average and low-filtered result (red) and the standard deviation of the results of all neural networks (blue). The x-axis represents the sample number. The graphs show the pitch of a head that is nodding (tilting his head up and down). As is clear from the graphs some networks do a better job in predicting the angle that others. This is highly dependent on the context.

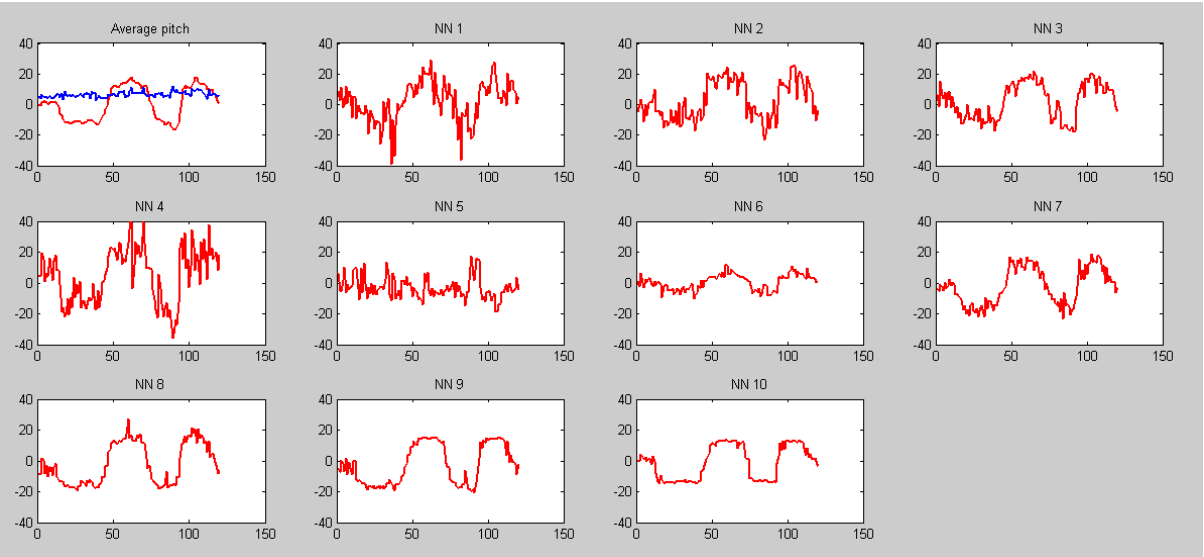

**Figure 5. Results Neural Networks**

## *The demo*

The program (see Figure 6) shows the grey-scaled webcam image (left window), the transformed image (right window, left panel) and the resulting estimated head pose (right window, right panel). The estimated gaze direction, without taking into account the eye-orientation, is indicated by the red arrow.

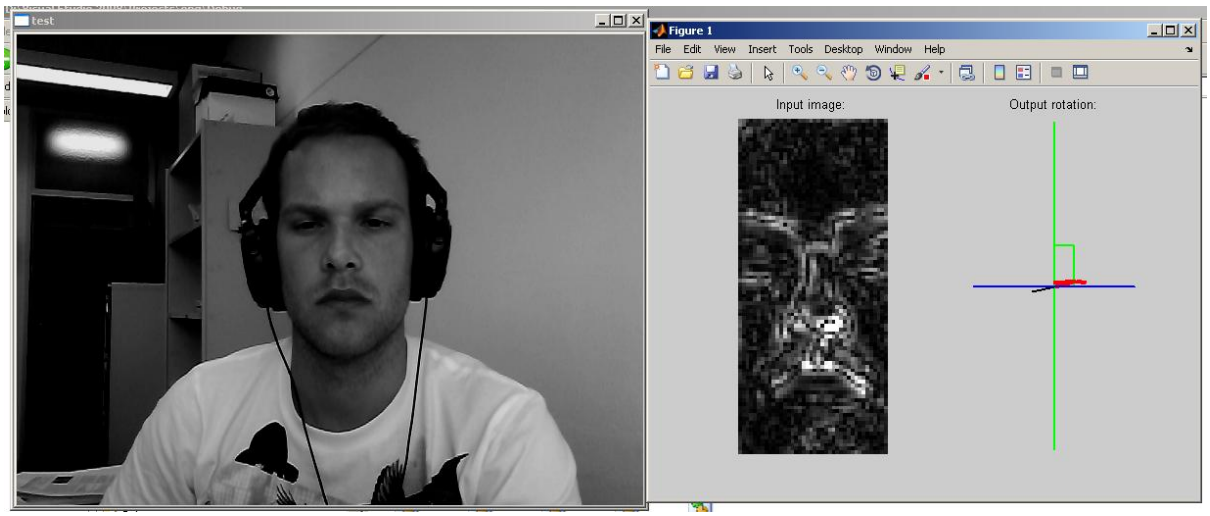


**Figure 6. Screen shot, head pose estimation program**

## *Conclusion*

The neural network approach provides a good estimate for the head pose. We still need to implement the networks in Nao and see how it performs. We did not test our network with new data. Tests still need to be performed to determine the accuracy of the method. Some progress can be made by adding neural networks trained with even more datasets. But also, the image can be analyzed for variables defining the context, e.g. lighting directionality, and these variable might then be used to select the x most optimal neural networks.

## *References*

Wollaston, W. H. (1824). On the Apparent Direction of Eyes in a Portrait. *Philosophical Transactions of the Royal Society of London , 114*, 247-256.

Kozima, H., Nakagawa, C., & Yano, H. (2003). Attention coupling as a prerequisite for social interaction. *The 12th IEEE International Workshop on Robot and Human Interactive Communication, 2003. Proceedings. ROMAN 2003.*, (Kozima 2002), 109-114. Ieee. doi: 10.1109/ROMAN.2003.1251814.

Viola, P., & Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple. In *Proc. IEEE CVPR*. Retrieved from http://doi.ieeecomputersociety.org/10.1109/CVPR.2001.990517.

Voit, M., Nickel, K., & Stiefelhagen, R. (2005). Neural Network-based Head Pose Estimation and Multi-view Fusion–Draft Version–. *Citeseer*. Citeseer. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.91.3945&rep=rep1&type=pdf.