

Gesture Recognition on AIBO

S. Radhakrishnan, W. D. Potter

Artificial Intelligence Center

University of Georgia

111 Boyd GSRC, Athens GA 30602, USA Phone: 706-542-0358

gopika@uga.edu, potter@uga.edu

Abstract

This paper describes a method of employing a learning algorithm to efficiently classify gestures made by a human to a robot. Such an algorithm acts as a more powerful communication interface for Human Computer Intelligent Interaction. AIBO classifies the gestures performed by a person using neural networks and then performs actions corresponding to the gestures. This application is designed to make AIBO more accessible as a robot attuned to human behavior.

Introduction

Gestures are expressive, meaningful body motions- i.e., physical movements of the fingers, hands, arms, face, head, or body with the intent to convey information or interact with the environment. There are three functional goals of human gestures, semiotic which refers to communicating meaningful information, ergotic which refers to manipulating the environment and epistemic which refers to discovering the environment through tactile experience [1]. In this paper we concentrate on the semiotic goal of gestures for humans to communicate effectively with robots. The use of human gestures has become an important research aspect of Human Computer Intelligent Interaction.

Sony AIBO was originally marketed as an entertainment robot but the abilities that the robot possessed in being a research tool led SONY to actively promote AIBO for academic research. AIBO was one of the first robots to change the entire perception of human-robot interaction. Its ability to act autonomously has been an important reason for its success. This paper aims to extend the existing behavior capabilities of AIBO to include gesture recognition.

It is important for robots to recognize objects based on features rather than color because it leads to better recognition capabilities. Many of the recognition capabilities of AIBO rest on its ability to distinguish colors and this aspect has been researched extensively in the Robocup series. This research paper extends existing work by training AIBO to learn gestures based on the features associated with the gestures.

Previous research in gesture recognition on AIBO has been done on a remote computer and the results have been beamed on to the robot through a wireless connection [2].

The research has been done on the same model of AIBO as ours. To create a truly autonomous behavior, we need to be able to perform the processing on board the robot instead of a remote computer. This research builds on our previous work with AIBO to recognize numbers and characters to solve mathematical expressions [3]. The results here are compared to the results obtained in the previous research as far as classification accuracy goes. This methodology can be presented to AIBO to pick up visual cues for various actions that it needs to perform. Our approach currently considers static hand gestures performed in a controlled environment with a constant background.

The rest of the paper is divided as follows: We describe the platform used to program AIBO in the next section. Section 3 justifies the choice of the learning algorithm used. Section 4 describes the Image Processing tasks that were performed on the image taken by AIBO. Section 5 explains the offline learning and online testing performed using neural networks. Section 6 explains how the recognized gestures are mapped to the corresponding actions that AIBO performs. We then provide a section to describe the results obtained with the behavior created. Lastly we discuss future work that can be done to extend this research.

Background – AIBO, OPEN –R and Tekkotsu

AIBO, as mentioned earlier, is an entertainment robot that has been used extensively for research in universities around the world. There are many features of AIBO that make it an extremely efficient robot. AIBO has a 384 MHz MIPS (million instructions per second) processor that can handle multiple tasks simultaneously without actually affecting performance. It also has a 32 MB onboard RAM. AIBO has a wireless card that can be used to connect to a PC and can transmit and receive data packets to and from the PC. The wireless connection is 802.11b and the PC and AIBO can be connected through a router or over a peer-to-peer network connection.

Behaviors created for AIBO are created on the PC and stored on a memory stick that is inserted into AIBO. The robot has 20 joints, 18 of which are PID (proportional integral derivative) joints with force sensing and 2 Boolean joints. It has 9 LEDs that are used to express the emotions of the robot. It has a CCD (charge coupled device) camera with a field of view that is 47.8° high and 57.6° wide,

resolutions are 208x160, 104x80, 52x40 and can take up to 25 frames per second. It has stereo microphones and can detect the direction of sound using them. To avoid obstacles, AIBO has an infrared distance sensor which has a range of 100-900 mm. It has pressure sensitive and Boolean (on-off) buttons and updates sensor readings every 32 ms. The most important feature of them all is that AIBO is programmable.



Photo Courtesy of Sony Electronics, Inc.

Figure 1: AIBO ERS220A

OPEN-R is a software development environment used to program AIBO with C++ as the programming language. OPEN-R programs are built as a collection of concurrently running OPEN-R modules which are essentially different simple actions combined to form one single complex behavior [4]. One complaint regarding the OPEN-R architecture is that it is not well documented. The sample programs given on Sony's website do not have inline comments on them to provide an understanding of the workings of OPEN-R. Access to sensors and actuators using OPEN-R still involves the use of some low level programming skills. This led to the development of the Tekkotsu environment.

The Tekkotsu framework was developed by Carnegie Mellon University using OPEN-R as its basic framework. Tekkotsu allows for creating behaviors that can interact with OPEN-R without the hassle of dealing with low level programming. The Tekkotsu framework provides access to all the sensors and actuators on AIBO by just invoking the necessary values and lets the user concentrate on creating behaviors [12]. It is possible to control AIBO wirelessly using a peer to peer network connection and the telnet console on a host computer or laptop. Tekkotsu has implemented a GUI called Tekkotsu Mon to remotely access the behaviors that run on AIBO from a laptop. This facilitates an easier debugging process and better PC-Robot communication.

The Learning Algorithm

Of the different learning algorithms that can be used (Bayesian Learning, Nearest Neighbor, Neural Networks, Decision trees and others), Neural Networks are most commonly employed in classification and pattern

recognition problems. This is because of their ability to generalize on fewer training examples and their robustness to error [5, 14]. Unlike other learning methods, Neural networks that have learned a particular task can be used on any system by simply porting the weights of the network to the system. This leads to saving memory on devices that may not have much memory to start with. Due to the potentially long training time associated with real time training, the actual learning process is conducted on a laptop and the results are saved on to the Memory Stick®. Since training and testing are to be conducted on different platforms, it is important to be able to transfer the trained results to the testing platform with ease. Since the trained results of a neural network are a set of weights, they can easily be transferred to AIBO. Neural networks work better than pure image processing techniques like template matching because the computational intensity of template matching is much higher than neural networks and it might overwhelm the processor on AIBO.

Our experimental goal is to have AIBO look at a hand gesture, take an image of it, process the image to send to the neural network and then let the neural network classify the image based on the existing set of learned weights for the different gestures. The neural network employed in this problem is a back propagation neural network with 400 pixel inputs, a 40 unit hidden layer and a 9 unit output. The 9 gestures that AIBO can recognize are just a sample of the number of gestures it can actually classify.

Image Processing

Image Processing used in the Robocup tournament emphasizes the need for color segmentation and object recognition using color recognition [13]. This is an extremely efficient method when considering specific environments that are defined based on color as with the robot soccer field. But under circumstances that do not pertain to the constraints of Robocup we need a different methodology to recognize different objects. There are two variations to the image processing done in this paper. One method processes grayscale images while the other method processes color images. The Sony AIBO comes equipped with a color CCD camera and is tuned to detect colored objects, such as the pink ball that comes shipped with it. The gestures are performed with yellow gloves when dealing with color images to facilitate easy detection. The gestures are performed with white gloves when considering grayscale images. The idea behind performing the same gesture recognition task using different image formats is to evaluate the difference in performance between using color images and grayscale images. The images taken by AIBO are 352x288 in size which is the largest sized picture that AIBO can take. The pictures are taken in the RAW format which is not modified by any compression algorithm. The image is initially taken by AIBO and then processed by the image processing algorithm.

The image processing is classified into three parts. The first part extracts the pixel information from the rest of the image (headers). This part is different for the grayscale and color images as explained later. This pixel information is then sent to the portion of the algorithm that detects bounding boxes. Bounding boxes are detected based on a threshold level. We assume in a grayscale image that the gesture is the brightest part of the image. In the case of color images the gesture is of a different color than the rest of the image. This does not lead to classification of the gestures based on color because we still classify the images based on features using the learning algorithm.

The third part compresses the image to a constant size which is then fed to the neural network. The image processing algorithm runs individually on every image that is saved, by creating a bounding box around the object of interest in the image, compressing the bounded object to a constant size and sending it to a neural network to be classified. This method works the same for both color and grayscale images. Figure 2 shows the different hand gestures that are considered in this research.

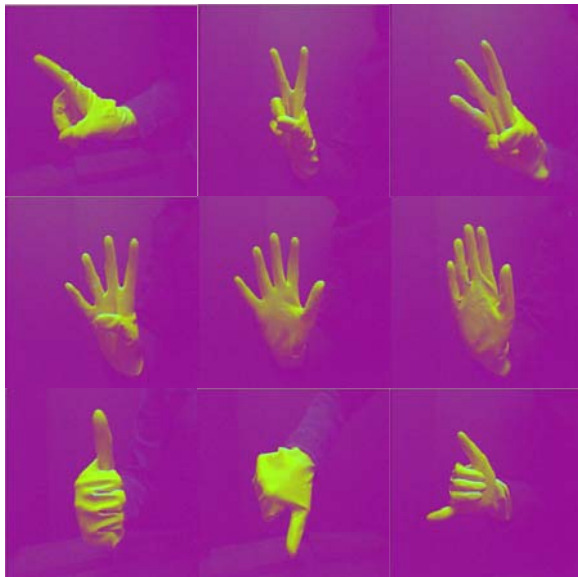


Figure 2: A sample set of the hand gestures

Pixel Extraction from color and grayscale images

Pixel extraction involves the removal of the header and other unnecessary features of images while retaining just the pixel values. The image saved to the Memory Stick[®] is in the RAW format which means the pixels are unchanged from the camera. The color format of the color image is YUV where Y stands for brightness, U and V stand for chrominance. U relates to the blue-yellow color components while V relates to the red-green color components of the color image [6]. Since the gestures are all yellow in color, it is enough to extract only the U component of the pixel. This method leads to fewer computations on the part of the image processing

algorithm since it does not deal with the other two dimensions of the color image. Since the V component contains color information, it is to a large extent independent of varying illumination. This method cannot be recommended for other recognition tasks since a large amount of information is lost by ignoring Y and U components. The V component is extracted and sent to the step where a bounding box is created. In the case of grayscale images, the pixel values all relate to brightness levels. The pixels are just extracted and sent to the bounding box segment which is explained next.

Edge Detection:

Instead of detecting the images using color and brightness threshold levels, an alternative is to detect the gesture by using edge detection algorithms on the image and comparing the edge detected images to a general hand template. Different edge detection algorithms were used on the image, like Sobel, Roberts and Canny [9]. The Canny edge detector gave the best edge detected image, based on noise levels in the modified image. But edge detection could not be used because when the edge detected image was compressed to a 20x20 size, it lost its shape properties. The reason for this behavior is the edges being too thin in the image. So when a 240x240 image is converted to a 20x20 image there are not many properties of the image that get retained. Hence this method was discontinued.

Creating a Bounding Box

Bounding boxes are created when the necessary parts of the image are to be segmented from the rest of the image. This way the neural network gets only the actual gesture as input which makes the classification process more accurate. A bounding box is created around the gesture based on brightness or certain color values. This is done by segmenting the yellow pixels from the non-yellow pixels and extracting the ones that contain information about the character. These bounding boxes are then converted into a constant size of 20x20.

Dimension Reduction

Neural Networks require a consistent number of inputs across all training examples [7]. This feature constrains the Image Processing algorithm to output a constant number of pixel values which is then fed to the neural network. A reliable number of input values to the neural network was found to be 20x20 or 400 by trial and error. Different algorithms were considered to perform the averaging process to convert a random sized bounding box to a constant 20x20. One algorithm that retained the original shape best was the simple averaging process. The bounding boxes formed for the characters were of arbitrary sizes and to make them all a 20x20 image, averaging operators were used. The bounding boxes were all padded with non-pixel values (typically 0) to fit one of 20x20, 40x40, 60x60, 80x80, 120x120, 160x160, 180x180 or 240x240 sized boxes. Then 2x2 or 3x3 averaging operators

were applied to them when necessary, to reduce them to 20x20. We did not encounter bounding boxes that exceeded 240 pixels in height or width. Hence sizes over 240x240 were not considered. The 20x20 image is the input to the neural network that classifies the images.

The inputs to the neural network retain the same shape as the original gesture as shown in Figures 3a, 3b and Figures 4a, 4b. Uniform Thresholding has been applied to the pixel values making them high (1) or low (0) where high is part of the gesture and low is the background [8, 9].

The Learning Process

The Learning algorithm is divided into two parts, Offline Learning and Online Testing. Offline here refers to learning taking place on a remote computer and not on the actual robot. It takes a prohibitively long time for the learning process to be performed onboard AIBO's processor. The advantage of having the robot perform real time learning is to better react to new circumstances or environments than what offline learning would allow. But real time learning also increases the interference effect associated with Neural Networks where learning in one zone causes loss of learning in other zones [10]. The training examples for the grayscale images have been taken under various lighting conditions and well represent the test samples that AIBO might encounter. The training examples for color images need not be taken under different lighting conditions since we consider only the V component of the image which is independent of brightness conditions. Hence the disadvantages that one might face by training offline are offset by the varied training examples. The neural network is not trained to detect rotational variance in images since gestures are viewed upright only.

Offline Learning

Offline Learning and Online Testing are carried out using Neural Networks. The reason for choosing Neural Networks is the ease of portability these systems possess. A learned system can be transferred to any robotic platform by merely transferring a set of weighted numbers. Another reason is that the Sony AIBO runs all the behaviors or applications stored on a Memory Stick[®] which is 16 MB in size. It is important that the Memory Stick[®] is not used up entirely for the learning process.

The simple back propagation neural network has been used extensively for classification systems and in pattern recognition [5, 11]. Hence this neural network was considered with 400 inputs which correspond to a 20x20 image as shown in Figures 3b and 4b. The number of hidden nodes was selected to be 40 based on trial and error. The system has 9 output nodes, one for each gesture performed. The different gestures are depicted in Figure 2. The input sent to the neural network is not scaled, since it has already been scaled to 0 or 1. The logistic activation function is used on the hidden and output layers.

The training set consists of grayscale and color images. The system considers grayscale and color images separately. A set of 30 images are taken as samples per gesture for each color scheme. This method looks to see which color scheme gives better results. The results are discussed in later sections.

Online Testing

Online Testing is a process of using the learned weights and classifying new instances based on these weights. The behavior on AIBO runs a simple feed forward neural network which takes the weights learned through offline learning and interprets the input gesture that is presented. AIBO serves as a platform for testing the results obtained from offline learning. The behavior runs the same image processing algorithm that is run on the offline learner. The gesture input is processed and sent to the neural network which classifies the input gesture and performs a certain action associated with it. This process is described in the next section.

Gestures to Actions

As a means of recognizing the gestures, AIBO performs certain distinct actions relating to the gestures which provide human interaction between robot and human. Tekkotsu provides us with different motion sequences that can be combined to allow for distinct actions to be performed. Sophisticated motion sequences can be created by dynamically creating a Motion Sequence Command. Different posture files (.pos) can be combined to form unique motions for the gestures. The different motions associated with the gestures are given below.

Gestures	Actions
One	Sit
Two	Stand
Three	Turn Left
Four	Turn Right
Five	Glance around
Stop	Freeze up
Thumbs up	Raise arm and wave
Thumbs down	Lower head and shake
Horns	Lie down

Table 1: Gestures and the actions corresponding to gestures

Results

Initially the gestures were performed with just the hand (without gloves on). This method proved to be difficult to expand since some hands are fairer than others and get recognized better. To avoid this effect, every hand depicting a gesture was clad in a white glove or a yellow glove to maintain constancy.

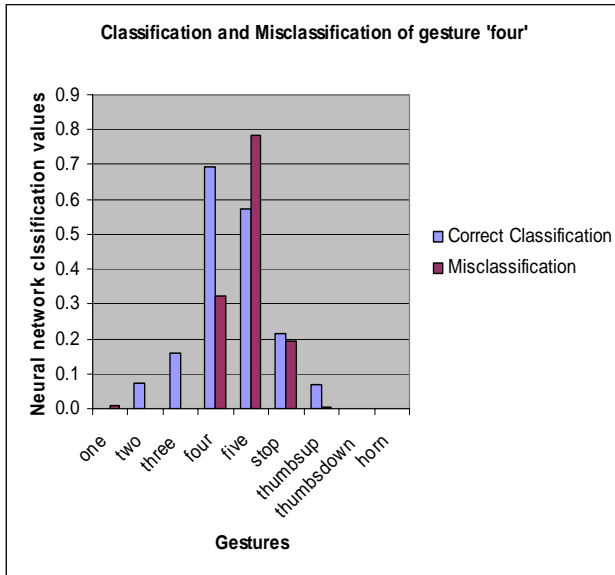


Figure 5: Classification result of gesture 'four'

Classification Results: The neural network classification of grayscale images gave much higher error rates than that obtained in the previous research [3]. The error rate of the classification increased because of the influence of brightness values in determining the hand gesture. If a shadow is encountered while performing the gesture there is the possibility of the gesture not being recognized properly. In spite of this disadvantage the results of the grayscale gesture recognition have been encouraging. The results of the color gesture recognition were comparable to those obtained with the previous research.

Figure 5 shows the correct classification and misclassification of gesture 'four'. The blue bars show the correct classification of 'four' as 'four'. Sometimes 'four' is thought of as 'five' which is shown by the red bars. The numbers are the neural network's classification result values for the gesture 'four'. The other gestures have either 0 or very low values for gesture 'four' because they are not confused with 'four'. This phenomenon brings down the accuracy levels to about 70% in grayscale images and to about 90% in color images.

Initially the system was trained on just one person's hand gestures. Later, the system was tested on different persons' hand gestures. It was observed that even though the system was trained on one particular hand's gestures, it could classify the gestures with the same accuracy for any hand. This performance can be attributed to the image being reduced and converted to binary and the fact that neural networks are able to extrapolate with few training examples.

Future Work

Our research attempts to achieve rotational invariance (to an extent) and view independent gesture recognition on AIBO. This method has worked well on individual images taken by AIBO. The research can be extended to incorporate noise in the environment. We currently use a constant black background to achieve a distinction between the gesture and the rest of the image. Expanding the training samples to include image sequences and dynamic gestures can be explored in future research. It is important to consider that AIBO is computationally limited with respect to implementing image processing algorithms. The reason we chose not to use image sequences was that it becomes too computationally expensive for AIBO to handle. But different techniques can be researched to reduce the processor intensive image analysis computations.

Gesture Recognition has been researched extensively on systems that have enough processor capabilities to run them. It is important to extend such research to platforms that allow for a powerful interface between humans and machines. This platform can be extended to successfully learn and analyze handwriting of different people. It has already been successfully tested on numbers and operators which were combined to solve mathematical expressions.

References

- [1] Cadoz, C. (1994) *Les Réalités Virtuelles*. Dominos, Flammarion.
- [2] M. Hasanuzzaman, T. Zhang, V. Ampornaramveth, Kiatisevi, P., Shirai, Y., Ueno, H., "Gesture based human-robot interaction using a frame based software platform", *Proceedings of International Conference on Systems, Man and Cybernetics (IEEE SMC 2004)*, pp.2883-2888, Hague, Netherlands, Oct. 2004.
- [3] S. Radhakrishnan and W. D. Potter, Basic Math via Character Recognition on AIBO Robot Dog, *Transactions on Information Science and Applications*, 1(3) pp. 127-133, 2005
- [4] OPEN-R SDK <http://www.openr.aibo.com>
- [5] Bishop, C.M., *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford UK, 1995.
- [6] Miano John, *Compressed Image File Formats*, Addison-Wesley Publications, 1999.
- [7] J. Ramesh, K. Rangachar, S. Brian G. *Machine Vision*, New York: McGraw-Hill, p.474-479.
- [8] Jahne Bernd, *Digital Image Processing*, Springer-Verlag, 2002.
- [9] Nixon, Mark S., and Aguado, Alberto S., *Feature Extraction and Image Processing*, Newnes Publications, 2002.

- [10] S. Weaver, L. Baird, and M. Polycarpou. An Analytical Framework for Local Feedforward Networks. *IEEE Transactions on Neural Networks*, 9(3), 1998.
- [11] Shalkoff, R.J., Pattern Recognition- Statistical, Structural and Neural Approaches, Wiley and Sons Inc., NY USA, 1992
- [12] Tekkotsu, <http://www.tekkotsu.org>.
- [13] RoboCup, <http://www.robocup.org>.
- [14] A. Grossmann and R. Poli, Continual Robot Learning with Constructive Neural Networks, *Proceedings of the 6th European Workshop EWLR-6 Brighton, England, August 1997*, pp. 95-109.