ISSN:0975-9646

Prathmesh More et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (1) , 2014, 534-537

Neural Network based Dynamic Kinesics Recognition for Gaming Interface

Prathmesh More, Abhijeet Thenge, Tushar Ghanti

Computer Department, SCOE,
Sudumbare, Maval, Pune.

***Abstract*— Kinesics recognition is an area of active current research in computer apparition. In this paper we focus on the problem of dynamic kinesics recognition using neural networks with the use of skin detection algorithm. A kinesics is defined as dynamic motion of hand to communicate with computer system. In this, we have given our hand outs on game playing but it can be made adaptive for any other applications.**

***Keywords*— Neural Network, Gesture Recognition, Gaming, Learning**

I. INTRODUCTION

This paper introducing a new concept of game playing wherein the same basic goal is achieved but rather in a different way i.e. instead of using the customary approach the game proceedings are controlled by human gestures. While the current input devices provide a limited set of values, the new approach can deploy commands with new values not possible for old devices. This will provide a more sophisticated interface through which the interaction with the machine will be of ease.

Interactive applications create picky challenges. Very fast response time. The no appreciable delay between users makes a gesture or motion and the computer responds should not sense by user. Not a lot have been changed for the most common Input devices of the computer since their introduction. We can consider the adequacy of the existing devices as one for the reasons for it. The most common means of communicating with a computer device are keyboard, mouse, scanners and a few more. Kinesics recognition has various applications like play and pause for video, games on computer, machinery control, static pose for the learning, and thorough mouse as well as keyboard and joystick replacement. Computer recognition of kinesics may present a more natural-computer interface, allowing people to point, or rotate a 3D model or draw pictures by rotating their hands. Hand recognition is currently in boom in the gaming industry. Rapid technological advancements have resulted to deploy games on various platforms. Various technologies have been implemented in the Gaming Environment which has resulted to more and more innovative ideas. Consoles used in the Gaming are improved day by day. In the computer interface two types of gesture are distinguished: Offline gestures-those gesture that are processed after the user contact with the entity. An example is activating a menu using gesture. Online gestures- manipulate gestures directly. These are use to scale or rotate a tangible object.

Until now there is no replicated way for the game playing using gesture recognition for the desktop platform. That’s what we are going to present in this paper.

II. AVAILABLE APPROACHES AND SOLUTION

In this kinesics recognition current input devices are wired gloves, depth-aware cameras, stereo cameras, controller based gestures and single camera. The approaches includes current work in the computer field on capturing kinesics or more general human styles and actions by interfacing with cameras as input device. The first gestures that were applied to computer interactions date back to the PhD work of Ivan Sutherland [11], who demonstrated Sketchpad, an early form of stroke-based gestures using a light pen to manipulate graphical objects on a tablet display. This form of gesturing has since received widespread acceptance in the human-computer interaction (HCI) community.

14 gestures in real time recognize by vision-based system to manipulate windows and objects within a graphical interface was developed by C.W. Ng et al. in [12]. Abe et al. [13] proposed a system which recognizes hand kinesics through the recognition of the bending of the one hand’s all fingers; depend on image-property investigation.

New. J.R. et al. [14] for track hand movement presented a real-time kinesics recognition method, define direction, and determine the number of fingers being held up in order to allow control of an underlying function.

In the work of Franklin et al. [16], a robot waiter is controlled by hand gestures using the Perseus architecture for gesture recognition. In the work of Cipolla et al., a gesture-based interface for robot guidance is based on uncalibrated stereo vision and active contours. The research of Guo et al. [14] discusses vehicles controlled by hand gestures, based on color segmentation. Waldherr et al. [15] proposed a vision-based interface that instructs a mobile robot using pose and motion gestures in an adaptive dual-color tracking algorithm.

Yin Xiaoming et al. [18] for hand segmentation used an RCE neural network based color segmentation algorithm, in which extract edge points of fingers as points of awareness and competition them based on the topological features of the hand, such as the midpoint of the palm.

III. SYSTEM DESIGN

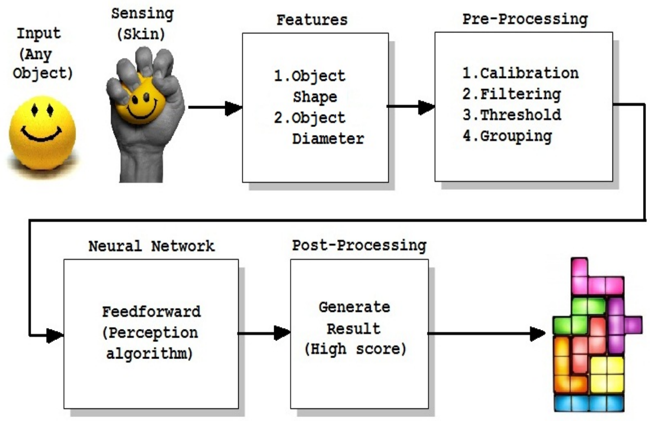

Fig. 1 System Flow

For achieve our goal we segregate system into three modules as follows:

A. *Kinesics Module*

Now we have to find a way that will be able to detect the human kinesics of the user. So that we need additional hardware support. To capturing the human hand signs we are using the camera. It is not sufficient to capture the human hand signs only so there is need of sign identification methodology. The method that should be capable of tracking and identify the different hand signs as well as the different hand skin colors. To identify the different skin colors is a difficult task because different location person have different skin colors. So to overcome this problem in our case we have used some kind of detection algorithm, which is best suited for our project. In our project the interaction of user with application is totally dependent upon the human gestures. So to overcome the above mentioned problem we used Skin Detection Algorithm.

The Skin detection algorithm is very robust in nature and it can fairly identify the different skin colors. This algorithm consists of major phases;

Calibration- We are calibrating the input device using samples of different skin colors of different locations. To identify more skin colors it requires more general approach, to achieve this it needs more complex model results in more computational power. Many location skin colors have major differences as well as their intensities are different. To improve the overall result we have studied new pixel scaling method for testing pixels based on calibration values. For identifying skin colors, simple normalization method is used to obtain saturation value ranges [0.0 to 1.0] and the scaling method is used to obtain tint value by using equation 1 and 2 respectively. We have obtained the par results for descents, black and white using only one skin chrominance model based on the saturation calibration minimal value.

$$S = (S - Min[S]) / (Max[S] - Min[S]) \quad (1)$$

$$T = (T - Min[S]) / (Max[T] - Min[S]) \quad (2)$$

Filtering- As seen that in complex background images the non-skin pixel area is greater than skin pixels. To reduce the computational efforts we used the filter that removes all non-skin pixels from the sample images captured by the camera and it evaluates the skin pixels.

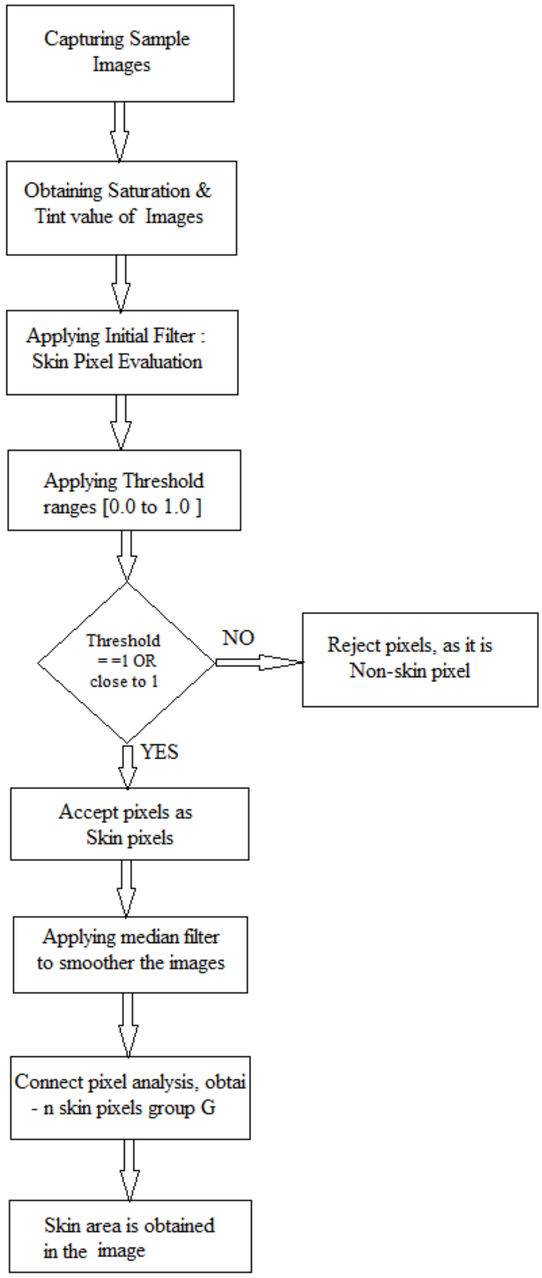

Fig. 2 Flow Diagram of Kinesics Module

Segmentation- Past studies shows that while using Gaussian model the TSL chrominance model is very effective for skin segmentation. It is true for where illumination condition varies. To obtain the equivalent pixel representation ST color space is used.

Threshold- Instead of using fixed threshold value we used different method where images are analyzed before binary selection. This approach is better where illumination variations take place and different skin colors appear. In this project the threshold method selects only part of image having threshold 1 or image pixels value close to 1. Using this method we can at least 100% match the non-skin pixels if there is no skin pixels present in the image.

Grouping- Using grouping method the images after the threshold processing are enhanced if looking for skin regions. In the grouping method we are applying the median filter to smoother images and to reduces the distorted isolated noise from the image. Median filter modifies small area of images and no skin pixel is erased. Connect pixel method is used to obtain the pixels group G, initially filtered by their size. To reduce some common errors we also join from the initial universe G for example smaller gap between two separate fingers. This was done by finding smaller expanse between groups that must be comparative to the size of one of the fundamentals. The grouping method is useful for hand and face gesture detection.

B. Neural Network Module

Artificial neural network is one type of network that sees the nodes as 'artificial neurons'. An artificial neuron is a computational model inspired in the natural neurons.

When modelling artificial neurons the complexity of real neurons is hidden. The activation of the neurons is strong-minded by inputs (like synapses), and multiplied by weights (strength of the respective signals), and then also computed by a mathematical function. Another function computes the output of the artificial neuron. Artificial neural network combine artificial neurons in order to process in sequence. Learning algorithms can be divided into two methods namely unsupervised and supervised methods.

Supervised learning is learning process that requires a set of labelled patterns (i.e. input patterns with known target outputs). Unsupervised learning is learning process that does not require desired response. The network determines the appropriate response to a given input.

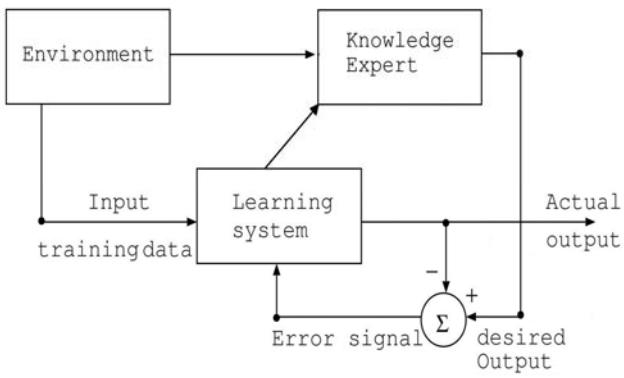

Fig. 3 Supervised Learning

Supervised learning is further classified into using reinforcement methods or error correction methods. Reinforcement learning is used when after each appearance of an input-output example we only know whether the system produces the preferred result or not.

1) The perceptron Algorithm:

Guided learning with augmentation method can be implemented by perceptron algorithm. A complete method to achieve a desired behaviour from the self-arranging computing units is perceptron learning algorithm. In this a desired response is achieved. This is done by simultaneously executing a correction step.

Hereafter for a couple of paragraphs we deal with the perceptrons learning strategies. We represent the following attributes;

i1.....in - input vectors (I/P to perceptron),
m1.....mn - weights of perceptron,
θ - threshold,
than;
m1.....mn+1 is extended weight vector with mn+1=-θ and i1....1 is the extended input vector.

By using the scalar product we evaluate the perceptrons threshold.

It is the context reference that shows whether we are using the extended vectors or we are using the normal vectors. The perceptrons mathematical test statistics are thus-

m.i>=θ for m and i as weight vector and input vector respectively, and m.i>=0 for m and i as extended weight vector and extended input vector.

We have used the perceptron algorithm in our project as follows:

We have an n-dimensional space of extended input. We check if there exists a vector m which can disperse both sets. Thus doing so we have two training sets, Po and Ne, and all the vectors mapped to open +ve half-space are in Po and all the vectors mapped to -ve half-space are in Ne.

The generalized steps are-

i) START: Generate the m0 at random. This represents the weight vector.
ii) SET: t=0
iii) TEST: An input vector i which belongs to Po and also belongs to Ne is selected in any order,
goto TEST if;
i belongs to Po and m.i>0
or i belongs to Ne and m.i<0
goto ADD if;
i belongs to Po and m.i<=0
goto SUBTRACT if;
i belongs to Ne and m.i>=0.
ADD: set t+=1 and m+1=m+i, goto TEST
SUBTRACT: set t+=1 and m+1=m-i, goto TEST

If the random vector that we pick from Po or Ne fails to get classified correctly than this algorithm makes appropriate correction in the weight vector. On correctly mapping all the vectors, the cycle can be stopped. We take care of the weight vector m to be updated only in limited iterations by using Perceptron convergence algorithm. The following test guarantees that the above perceptron algorithm will stop and thus make it a reliable.

2) Convergence of the algorithm:

From the cases given below we can see the ease of convergence proof of the perceptron learning algorithm. The weight vector m is updated by the perceptron learning algorithm. This happens only when the sets Po and Ne are measurable and are linearly distinguishable.

The algorithm converges if:

i) Po~ = Po+Ne`, where;
Po~ is the set formed by union Po and Ne,
Po is the +ve set,
Ne` is the set have vectors of Ne in negated form.

ii) If we consider that there is a solution for the linear disintegration problem than we call m` the normalized solution vector.
iii) If m.i>0 for a weight vector m, than Po can be normalized. Note that this is true for ηi, where i is the input vector and η is a constant.

If the inputs are managed in a messed up way and I/P vectors aren't normalized than the computation may extend to a large number of steps, thus making it a tough task for perceptron learning algorithm to converge to a solution.

C. *Application Module*

Here we refer to our module as application module but it can be use in this paper as game to which our concept is being implemented.

We can take any game, but the thing to remember is that more complex the game is more kinds of gesture need to handle. For start work we will take the simple game of tetris where we have to move the block of different sizes in a rectangular play area. As well as we tried for Ping-Pong. If we are able to make one complete line with the help of these blocks we score points. Difficulty of the game increases level by level i.e. the speed of the arriving blocks increases.

In these we have deal with four gestures types 1. Move the block to left 2. Move the block to right 3. Move the block to downwards 4. Upward for rotate the blocks.

We can have various other gestures depending of our games like if the take the example of complex games like Call Of Duty we will have gesture for shooting with the gun and rotating our viewpoints, movements etc.

IV. CONCLUSIONS

This paper give you the initiative how the kinesics recognition can be done using the neural network for games like Tetris and Ping-Pong. For the implementation of various applications we have given the basic introduction to neural network. We will try to implement our concept as mentioned in the paper. For fastest way to achieve results c/c++ implantation will be preferable but this will lead to a huge complexity in the design which is not best suited and would be chosen to implement of neural network module, application module in c# and kinesics module in MATLAB. We want to work on the fields which will make our concept more consistent and fast.

ACKNOWLEDGMENT

We wish to thank Pro. Sunil Yadav and Pro. Palavi Jha for their valuable guidance and inspiration for this paper. We like to express a deep sense of gratitude from the bottom of our heart to our parents & well-wishers’, without whom it was impossible for us to reach at this stage.

REFERENCES

[1] “Hand Sign Recognition Through Palm Gesture And Movement“, Oleg R., Matt M., Springs 2012.
[2] “Study On Gesture Recognition System Using Posture Classifier And Jordan Recurrent Neural Network”, Hikawa H., Araga Y., IEEE.2011.6033250.
[3] “Design of Gesture Recognition System For Dynamic User Interface”, Rautaray S.; Agrawal A.; 10.1109/IEEE.2012.6208628.
[4] “Hand Gesture Recognition Algorithm Based On Grayscale Histogram Of The Image”, Igorevich, R.R.; Pusik Park; Dugki Min; IEEE.2010.5612056.
[5] “Artificial Neural Networks an Introduction”, K.L.Priddy, P.E.Keller.
[6] “Robust Skin Color Detection and Tracking Algorithm”, Patil P.; Patil Y.; IJERT, 2012.
[7] Duane Hanselman, Bruce Littlefield, “Mastering MATLAB, A comprehensive tutorial and reference”, Prentice Hall.
[8] “Artificial Neural Networks for Beginners”, Carlos Gershenson.
[9] “Artificial Neural Network Approach For Hand Gesture Recognition”, Yewale S.; Bharne P.; IJEST, 2011.
[10] http://www.dcs.gla.ac.uk/~jhw/nn.html
[11] Sutherland, I. E. Sketchpad: “A man-machine graphical communication system.” In: Proceedings of the AFIPS Spring Joint Computer Conference 23. pp. 329–346, 1963.
[12] C. W. Ng and S. Ranganath,: “Real-time gesture recognition system and application.” ImageVis. Comput., vol. 20, no. 13–14, pp. 993–1007,2002
[13] K. Abe, H. Saito, and S. Ozawa,: “Virtual 3-D interface system via handmotion recognition from two cameras.” IEEE Trans. Syst., Man, Cybern. A, vol. 32, no. 4, pp. 536–540, 2002.
[14] New, J.R., Hasanbelliu, E. and Aguilar, M.: “Facilitating User Interaction with Complex Systems via Hand Gesture Recognition.” In Proceedings of the 2003 Southeastern ACM Conference, avannah, GA., 2003.
[15] V. Ampornaramveth, Md. Hasanuzzaman, Tao Zhang, M.A. Bhuiyan , Y. Shirai and H. Ueno,: “Real-time Vision-based Gesture Recognition for Human Robot Interaction.” In the Proceedings of the IEEE International Conference on Robotics and Biomimetics, Shenyang China ,2004
[16] M. J. Swain, D. Franklin, R. E. Kahn, and R. J. Firby: Happy patrons make better tippers-Creating a robot waiter using Perseus and theanimate agent architecture. In Proc. Int. Conf. Automatic Face and Gesture Recognition, Killington, VT, pp. 14–16,1996
[17] C.M. Fu, F. S. Chen, and C.L. Huang,: ”Hand gesture recognition using a real-time tracking method and hidden Markov models.” Image VisionComputer, pp. 745-758,2003
[18] Y. Xiaoming and X. Ming,: “Estimation of the fundamental matrix from uncalibrated stereo hand images for 3D hand gesture recognition",Pattern Recognition, vol. 36,2003