IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 3, Ver. I (May-Jun. 2014), PP 56-63
www.iosrjournals.org

A Novel Approach to Design a Customized Image Editor and Real-Time Control of Hand-Gesture Mimicking Robotic Movements on an I-Robot Create

Soumyajit Ganguly [(1)], Satyajit Bhowmick[(2)], Arnab Pal[(3)], Sauvik Das Gupta[(4)]
[(1)(2)] *Department of Electronics & Communication Engineering, West Bengal University of Technology, Kolkata, India*
[(3)] *Department of Computer Science, Burdwan University, Burdwan, India*
[(4)] *School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK, USA*

***Abstract**: Image processing and computer vision are considered as one of the most promising as well as exciting domains of modern day engineering. Equipped with concepts of artificial intelligence and some very advanced algorithms and functions to deal with challenges of processing of images as well as to mimic gestures, these domains of engineering prove to be one of the star attractions of highest level engineering and technology. This paper deals with some very basic image processing functions and algorithms and the concept of gesture mimicking robots. Matlab based customized image editor can be made quite easily by enthusiasts of image processing and a robotic platform called 'iRobot Create' can be used to perform gesture imitations.*
***Keywords:** Image editor, gesture recognition, MATLAB, iRobot Create.*

I. Introduction

This paper can be divided into two parts - Part1 and Part 2. In Part1 of the paper the authors show a GUI based Image Editor built in MATLAB and some unique image blending and filtering functions to get artistic effects out of images. Part 2 deals with controlling an iRobot using gestures which are detected using image processing techniques in MATLAB.

Design of a MATLAB based image editor deals with blending of the functions of MATLAB image processing toolbox along with some algorithms made on our own. 'Digital Image Processing using MATLAB'[1] comes very handy on that front. Mainly users deal with three types of images. Binary images consist of only zero and one, zero being black and one being white. Grayscale images consist of values ranging from 0 to 255 with 255 being the brightest pixel value. These two types are represented by a 2 dimensional matrix where the total rows correspond to the height of the image and columns correspond to the width. Color images however are represented by a 3 dimensional matrix where there is a separate matrix for red, green and blue color channel each. Inside each channel there are 2-dimensional matrices. Now any mathematical operation can be performed over a matrix or even over combined matrices i.e. images which will be presented in the later sections of this paper. The image editor is equipped with basic image processing functions and some special editing operations. The user can customize the editor as per his/her requirements and can make it more & more advanced. On the other hand, in case of gesture recognition and imitation using iRobot, at first several processing functions are performed on the 'True color' (RGB) images which are captured by the webcam and then the iRobot is used to imitate those gestures. Wang et al. [2] developed an object detection method combining Recognition and Segmentation. We have tried to build some algorithms to make the robot mimic the gestures. Also 'Dictionary of Computer Vision and Image' [3] is a nicely written tool to inspire others to work on those fronts. In the next portion of the paper we will discuss about all these different aspects.

II. Image Editor

In this part of the paper, the Image Editor is discussed along with some unique functions. This editor is created using the Graphical User Interface (GUI) option available in MATLAB. The whole editor can be described by the tree as shown in Figure. 1. The Filters, Basic adjustments, Histogram and Edge detection are done using straight forward MATLAB inbuilt functions directly applied to the input image. Some of the other features used in the Image Editor are the Blend Modes and Effects which are discussed in detail below. These operations are performed in a per pixel basis without the use of inbuilt function.

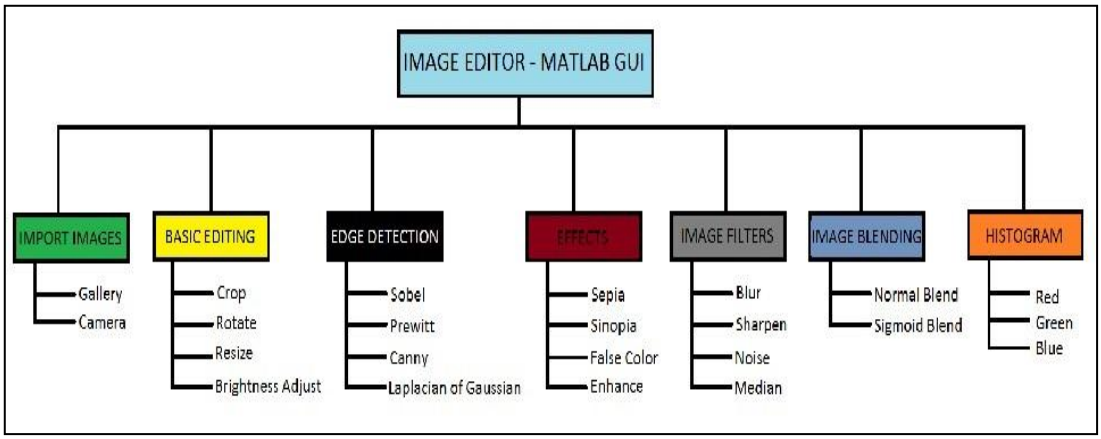


Figure 1. Function Tree

A. Effects

Effects consist mainly of image pixel manipulations which make any image look better or artistic. Some of these functions are applied on a per pixel basis, and others are based on image histogram analysis.

1) Sepia tone: Each and every pixel of the input color image is in the range 0 to 255. First these intensity values are converted to floating point. Now the intensities are in the range 0.0 to 1.0. For sepia tone conversion the formula used is shown in equation(1) where Rout, Gout and Bout is the corresponding red, green and blue values for output pixels and Ired, Igreen and Iblue are the input pixel values for red, green and blue channels.

$$Rout = 0.393\,Ired + 0.769\,Igreen + 0.189\,Iblue \quad (1a)$$
$$Gout = 0.272\,Ired + 0.534\,Igreen + 0.131\,Iblue \quad (1b)$$
$$Bout = 0.349\,Ired + 0.686\,Igreen + 0.168\,Iblue \quad (1c)$$

2) Sinopia: The exact same initial procedure is followed as in Sepia (A1). The final output color formula is different and is calculated as

$$Rout = 0.796\,Ired + 0.102\,Igreen + 0.102\,Iblue \quad (2a)$$
$$Gout = 0.254\,Ired + 0.470\,Igreen + 0.271\,Iblue \quad (2b)$$
$$Bout = 0.043\,Ired + 0.075\,Igreen + 0.500\,Iblue \quad (2c)$$

where Rout, Gout and Bout is the corresponding red, green and blue values for output pixels and Ired, Igreen and Iblue are the input pixel values for red, green and blue channels.

2. (a)

2. (b)

2. (c)

Figure 2: (a) Original image, (b) Sepia image & (c) Sinopia image

3) Enhance: Enhancing a particular image is mainly done by adjusting the contrast levels. This is achieved by obtaining the histogram of the image which is the pixel frequency distribution and then spreading out this distribution over the entire range of 0 to 255. A local adaptive histogram equalization method [4] is used which works for grayscale images. In this paper the authors present a way to extend this technique to color images. The input color image is in RGB format which is then converted to the HSV format. Now instead of red, green and blue the three color channels are hue, saturation and value. Individual histogram equalization is done on the

Figure 3. Enhance

saturation and value channels. Then the image is converted back to RGB color space. Adaptive histogram equalization is used for dealing with excessive noise due to image compression. The resultant image is highly enhanced as shown in Figure. 3, especially if the input image is washed out or under saturated and taken from an old lens.

B. Blending

We show blending of two images by three different types of blend modes. These modes use non-linear functions [5] for mapping the two images into output. There are parameters in the effects which can be tuned by a slider in the GUI.

i) Normal Blend: The first mode is simple averaging where each and every pixel of the first image matrix is averaged with the exact corresponding pixel of the second image matrix. The averaging factor is kept 0.5 to be default which results in a uniform mix between the two images. Changing the slider however will result in one image being mixed more than the other in the output.

ii) Sigmoid Blend: Another type of blend mode is shown by us which results in a rather interesting mix of two images. The image starts with the first image but gradually changes over to the next. The transition from one image to another can be controlled by a slider. This effect can be seen in the GUI **(**Figure. 4). The general formula for blending two images is done by applying a mathematical formula for each and every output pixel.

$$Iout = \alpha I1 + (1-\alpha) I2 \qquad \text{where } 0 < \alpha < 1 \qquad (3)$$

The value of α in (3) remained a constant 0.5 for normal blending. In sigmoid blending however the value of α changes throughout the width of the image. The change is sigmoid in nature, thus the name and is shown in equation (4).

$$\alpha_i = 1 / (1 - e^{-i}) \qquad \text{where i = k.j} \qquad (4)$$

The value of k in (4) is adjustable by the blend slider. This can give the user a customization over the degree of transition. Value of j in (4) is the current column. Replacing j with the current row would result in a horizontal blend rather than a vertical blend.

iii) Gaussian Blend: Gaussian blend performs similarly to the sigmoid blend function. The main program flow and algorithm remains the same except for the sigmoid function. In this type of blending the change is a Gaussian bell shaped curve in nature and is implemented using the normal distribution as shown in equation (5).

$$\alpha_i = (1/\sigma) . \ e^{-(i-\mu)2} \qquad \text{where i = k.j} \qquad (5)$$

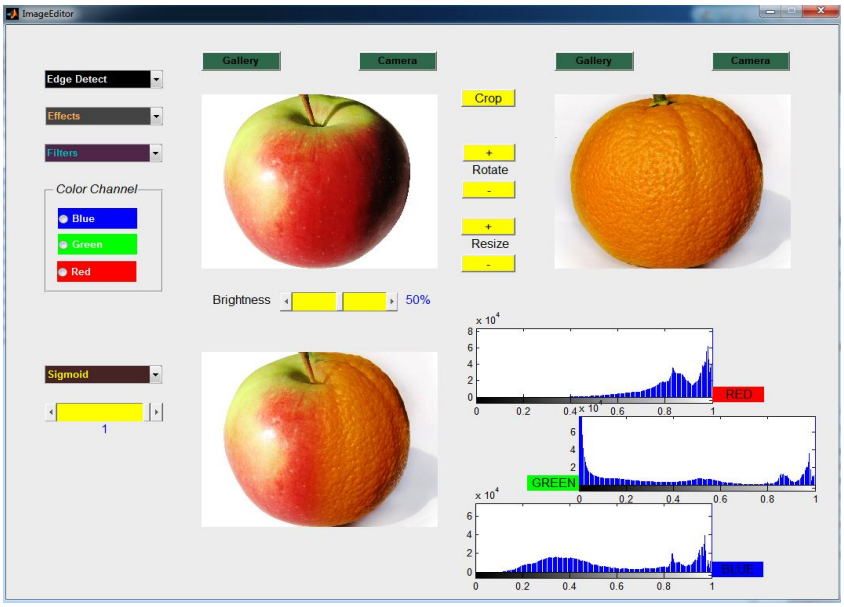


Figure 4. The result of Sigmoid Blend

The values of σ and μ in (5) are kept a constant depending upon the height and width of the input images.

It is to be noted that the second input image is being resized to the width and height of the first input image before any blend operation is performed. This is because blending works only with matrices of similar dimensions.

III. Hardware Platform

In this section, how different geometric shapes can be imitated using image processing and computer vision is detailed. For physical realization of the imitation simple hardware platform has been used that consists of

a. Image capturing device

b. iRobot create
c. Bluetooth Adaptor Module(BAM)

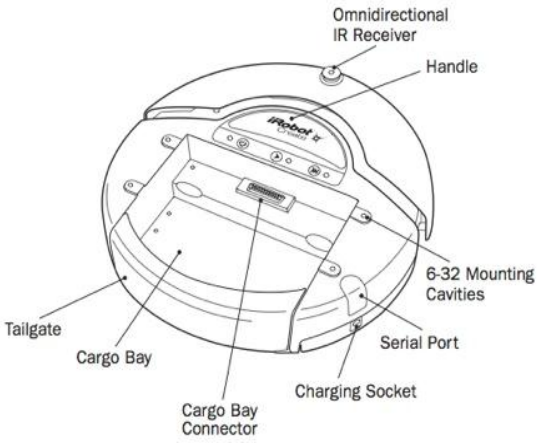
A. Image Capturing Device

Webcams are used as image capturing devices here. If there is no in-built webcam in the laptop or computer then external webcams can be attached to it to capture images. Continuous snapshots of the object, whose movements we want to imitate of, are taken. It looks like a continuous video stream due to human's perception of vision.

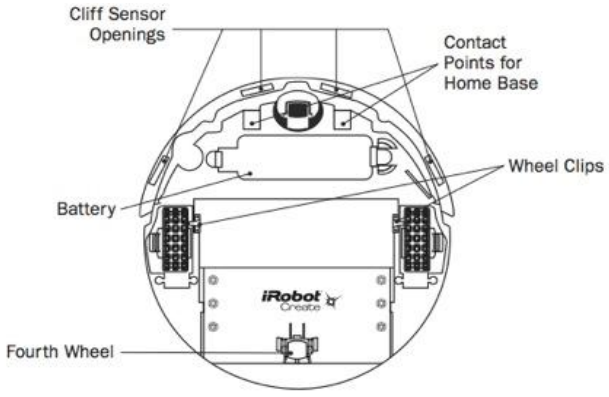
B. iRobot Create

'IRobot Create' is an affordable, preassembled mobile robot platform that provides the opportunity to program behaviors, sounds, movements and add additional electronics. It is manufactured by 'iRobot' that is based on the Roomba platform and was introduced in the year 2007. The iRobot Create includes a cargo bay which houses a 25 pin port that can be used for digital and analog input and output. The Create also possesses a serial port through which sensor data can be read and motor commands can be generated using the "iRobot Roomba Open Interface protocol"(a MATLAB based function dedicated to control iRobot Create). iRobot Create comes equipped with wheel clips that hold its main wheels in the retracted position. We can remove the wheel clips, which automatically places the wheels into the released position. iRobot Create comes with an additional unattached fourth wheel that allows for greater stability and prevents the back of the robot from dragging when Payloads are added.

5(a)



5(b)



5(c)

Figure 5. (a).iRobot Create (b).Top view of the robot and (c).Bottom view of the robot

Matlab Toolbox for the iRobot Create lets us control the iRobot Create directly from a PC or laptop running Matlab (by Esposito and Barton) [6]. The iRobot create simulator is a MATLAB toolbox designed to visualize the robot's movement in different environments. The simulator is designed to accept autonomous programs written using the MATLAB Toolbox for the iRobot create.

C. Bluetooth Adaptor Module (BAM)

If we plan on mounting a computer to the robot itself, the serial cable (serial wired communication) will prove cumbersome as soon as the robot begins to move. To get rid of this, the robot needs to go wireless. To make the robot move smoothly the Element Direct Bluetooth Adaptor Module has been used that enables wireless control of the iRobot Create from any Windows, Mac or Linux computer, or any other Bluetooth enabled device. The BAM connects to the Create's cargo bay port - without any extra wires or cables. The BAM provides a virtual serial port connection between a Bluetooth host and Create. A PC can communicate with Create in the same way it would as if it were attached with a serial cable. The BAM gives a user complete wireless control of Create. It uses the Serial Port Profile (SPP) to provide a means to send and receive serial packets to the iRobot Create mobile robot. It brings flexibility in operation and allows the Create robot to be driven remotely with a PC hosted web server. The user can communicate with the iRobot in almost real time while running complicated algorithm on a host computer using this module.

Figure 6. Bluetooth Adaptor Module

Table 1. Bluetooth Specifications

Bluetooth Specifications	
Device Name	Element Serial
Available Services	SPP
RF power	Class 1 Bluetooth
Baud rate	57600
Data Bits	8
Stop Bits	1

Table 2.BAM specifications

BAM SPECIFICATIONS	
Operating Frequency	2.4GHZ
voltage	5V
Current	100mA maximum
Internal Antenna Multilayer Chip, Peak gain	0.5dBi
Operating range	91 meters
Operating temperatures	0 to 50 degree centigrade
Size	55x55x16mm

IV. Colour Based Gesture Detection & Visualisation Using iRobot

In this part, it will be shown how different object's movements as per geometric shapes can be recognized and then visualized using iRobot. At first, gestures are detected and then iRobot is accessed via Bluetooth to imitate those gestures. The following part details the procedure.

A. Color based Image Segmentation

The camera is initialized. Each frame is taken out from live video stream and converted to RGB color space. A loop is set up which iterates through each of the frames of the input stream. First the respective color channel is taken out from the RGB and stored in a separate matrix. Then the Input frame is converted to grayscale. An absolute difference image is created from the grayscale image and separate channel image. This resultant image is then segmented by histogram based method [7]. Now the output binary image has the colored object in concern as a blob of white pixels in a black image. The central moments of this image is calculated from which the centroid is kept in record for each frame. Noise is discarded by rejecting blob area sizes of less than 3000 pixels and performing morphological operations. Noise is further reduced by pre-calculating the threshold level in segmentation which is run for 30 frames in the beginning and the median value of 30 levels is used. This also greatly improves overall performance as the level is not calculated each and every time after the first 30 frames.

7(a)

7(b)

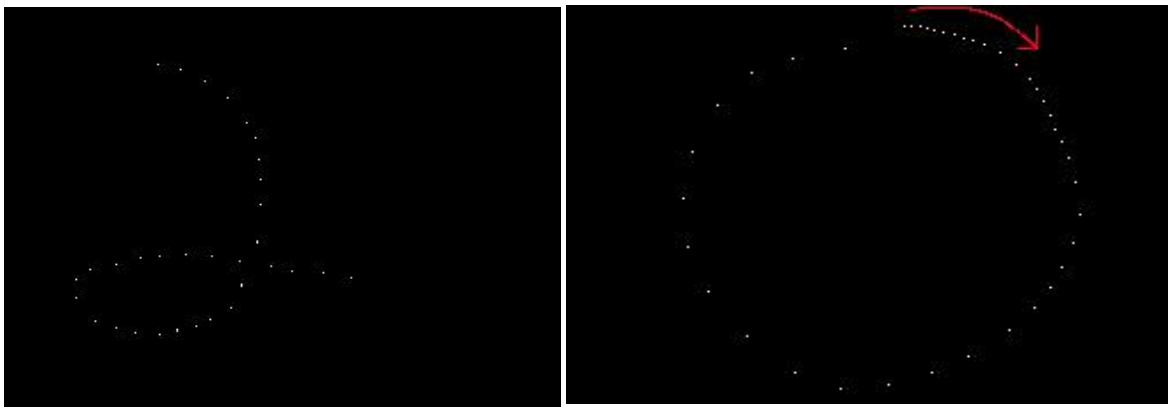
Figure 7. (a). Input video frame (b). Segmented out binary image from the frame

1) Square Gesture - The frames are segmented as described and the centroids per frame is taken into consideration. A distance window is estimated at about 30 pixels experimentally and a time window is kept as 5 frames per second. Now a running window of centroids is considered over the time window. If the total horizontal displacement is found out to be greater than or equal to 30 pixels between the first and last frame inside the running window, then a valid left or right movement is considered. Vertical displacement of the centroids gives the up or down movement. Now a succession of up, right, down and left is considered as a square gesture. The order is hard coded to be from up and can start from any direction. However only clockwise movement is considered.

2) Rectangle Gesture - The procedure is very similar to the square gesture detection but here two separate distance windows are estimated as 30 pixels and 50 pixels. Horizontal limit is 50 and vertical is 30 pixels. This process results in detecting a rectangle.

3) Triangle Gesture - This procedure is slightly different. The distance window is fixed at 30 pixels but here both horizontal and vertical displacement is considered at the same time for getting a sloped movement of the centroid. When 3 consecutive sloped movement is detected, the corresponding order of direction is checked. If condition is matched, a triangle gesture is detected.

4) Circle Gesture - For circle gesture detection, an entirely new method is proposed. The running window size is taken as 30 frames and distance window is kept at 40 pixels. For each new frame we have a history of the previous 39 centroid locations plus 1 present centroid location. The probable mid-point of these centroids is calculated. This is done by taking the top most point and bottom most point and passing an imaginary straight line through those points. The equation of the line is calculated by the 2-points form. Another line is calculated which joins the left-most point to the right-most point. Now the intersection of these two lines gives a rough estimate about the center of the 40 points. Euclidean distance from each point to this new center is calculated which gives a rough estimate about the radii. A statistical analysis of these 40 radii gives us a good idea of the gesture being drawn.



8(a) 8(b)

Figure 8. A snapshot of the rolling window of 40 centroids plotted in an image (a) Not detected as a circle (b) Positive detection of a circle

If the average radius is below a threshold of 25 pixels, it is considered as no-movement static noise and discarded. When average radius is higher than threshold but the variance is high then the gesture is not a circle. When the variance is within 20 percent of the mean, we get almost perfect circles being drawn by the colored object. After a particular circle has been detected, a complete window of the immediate next 40 frames is discarded to prevent over detection. This method works for both anti-clockwise and clockwise circles. There is no need to start the gesture from a particular point.

B. iRobot interfacing: iRobot is interfaced with computer wirelessly. For this purpose Bluetooth Adapter Module (BAM) is used and connected to the cargo bay of the iRobot. After completing the Bluetooth setup the MATLAB program is executed on the host PC. The iRobot is accessed via Bluetooth. Then Matlab Toolbox for the iRobot Create is used to control the robot and to imitate different shapes. The procedure is same for all the shapes, the only thing that changes is the commands which we use, defined in the toolbox, based on our desired operation.

V. Discussion & Conclusion

In this paper, some practically tested models and procedures have been depicted to build a customized image editor to experience the magic of image processing at the very basic level and also to have some fun by controlling a robot, imitating gestures. The image editor has performed operations very precisely and smoothly. In case of iRobot, the results that we have got are really almost precise, exciting as well as inspiring, given that there will always be some hardware limitations and real time transmission delay while operating the iRobot.

We want to extend our research work further to imitate more and more complex gestures and shapes. The shape that we have already experimented with is ‘eight’ (8). Though the iRobot operates fine imitating the shape, still it needs some fine tuning. We are working on that and hope to come up with this one and some more complicated gesture imitating robotic movements very soon.

References

[1]. Rafael C. Gonzalez, Richard E. Woods and Steven L, Eddins, “Digital Image Processing using MATLAB”, (2004).
[2]. Liming Wang, Jianbosh Shi, Gang Song, I-fan Shen, “Object Detection Combining recognition and Segmentation”, Fudan University, Shanghai, PRC, 200433.
[3]. R. Fisher , K Dawson-Howe, A. Fitzgibbon, C.Robertson, E. Trucco, John Wiley, “Dictionary of Computer Vision and Image Processing”, 2005.
[4]. H.D. Cheng and X.J. Shi, “A simple and effective histogram equalization approach to image enhancement“, Digital Signal Processing 14 (2004) 158–170.
[5]. T. Alan Keahey, Edward L.Robertson, "Techniques for non linear magnification transformations", Department of Computer Science, Indiana University
[6]. http://www.usna.edu/Users/weapsys/esposito/roomba.matlab/
[7]. Jun Zhang, Jinglu Hu, “Image Segmentation Based on 2D Otsu Method with Histogram Analysis”, JSPS Research Fellow Waseda University, 2008