Controlling Robotic Arm Using Image Processing

Authors : Roma Jain, Jenita Jeyakumar, Sameer Kadam, Nanditha Nirmal
Guide: Prof S.B. Deshmukh

Introduction

The world today is driven by machines of various kinds. Machines assist humans in performing different tasks ranging from simple to more complex ones & they are an indispensable part of our lives. A Robot is defined as a machine that is capable of performing a variety of often complex human tasks on command or by being programmed in advance. It is basically a mechanical arm which imitates human movement in a 2-D space. The hand movement is captured through a camera & it is converted into angle. This information is sent to the arm which move accordingly in order to imitate the human arm movement.This technology can be used in hospitals in case of emergencies when the doctor isn't available on a short notice.It's especially helpful in ambulances where time is critical. If implemented in the correct manner, this technology can improve the human lifestyle tremendously.

Objective

Imitation is the ability of an agent(living or artificial) to observe the actions of another agent and try to act like it. We have programmed the robotic arm to imitate the human hand movement. The first problem encountered by the robot is who/what to imitate. For this we use skin color detection so that only the moving hand is focused upon and rest of the unwanted information in the picture is removed.

Methodology

Skin color detection: YCrCb or YCbCr:The YCrCb stands for Luminance,Red-difference & Blue-difference chroma components. Firstly, for skin color detection we need to know the skin range to tell it apart from the surrounding. This range varies with the light intensity hence after taking a number of samples, we took it as :

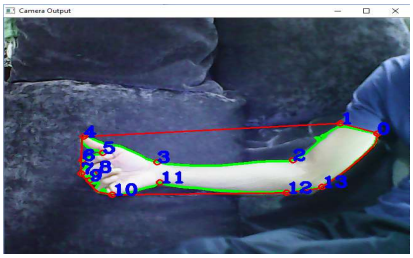
min_YCrCb= np.array([0,133,77],np.uint8)
max_YCrCb=np.array([255,173,127],np.uint8

For color conversion, we use the function : cv2.cvtColor(Image,flag) where flag is conversion type.

After conversion, cv2.inRange function is used which returns a binary mask, where white pixels (255) represent pixels that fall into the upper and lower limit range and black pixels (0) do not.

Contour formation:A contour refers to the outline or silhouette of an object. cv2.findContours(image, OutputArray contours, OutputArray hierarchy, int mode, int method)
Now to detect the hand, we need to know the approximate area of the contours formed. If the area is smaller than it, then it is not considered. After a number of samples and appropriate distance between the camera and the user, we found out that areas greater than 10000 is considered.Area of contour is same as number of pixels inside the contour. It can be found out using cv2.contourArea() function.

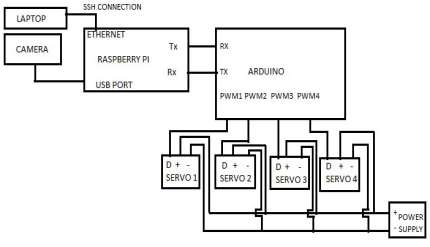
Approximation of the contours formed: Contour Approximation will remove small curves, there by approximating the contour more to a straight line. This is done using cv2.approxPolyDP() function which basically reduces number of points to operate.
approx=cv2.approxPolyDP(cnt,epsilon*cv2.arcLength(cnt,True),True)



Implementation

Hardware:

- Raspberry pi 2 Model B
- Arduino Mega 2560
- Servo Motors
- Camera



Mechanical Design

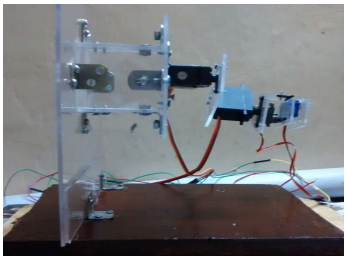


TABLE 1: TORQUE CALC. OF SERVO MOTORS		
Sr. No	Position	Torque
1	Shoulder	12kg-cm at 6v /9.8kg-cm at 4.8v
2	Elbow	12kg-cm at 6v /9.8kg-cm at 4.8v
3	Wrist	5 kg-cm at 6v /3.5kg-cm at 4.8v
4	Gripper	2.5 kg-cm at 6v /1.8kg-cm at 4.8v

Result and Discussion

Calculation of torque of the servo motors placed at various joints and material to be used for construction were the crucial part of our project. We had kept the length of the acrylic sheets as 15 cm but on practical implementation we found that servo motors placed at shoulder point caused a voltage drop in the circuit because of the heavy weight which could not be sustained by 12kg-cm torque motor. Finally the lengths were reduced accordingly. Calculation of skin ranges under various light intensities along with the minimum contour area had to be done for near-accurate results. Lastly the serial communication between the two microcontrollers with appropriate value of the time delay was the important part. Since servo motors need 15 milliseconds to reach a particular position according to the angle sent, the delay between two camera frames had to be increased to 0.1s.

Future Scope

Our future scope is to implement the arm in three dimensional space and use these methods for designing a humanoid capable of imitating the full human body movements.. They can replace human workers in harmful, potentially dangerous environments

References

http://docs.opencv.org/3.1.0/dd/d49/tutorial_py_contour_features.html#gsc.tab=0