INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH & DEVELOPMENT

ISSN 2278 – 0211 (Online)

Touchless Human-Computer Interface for In-Car Devices

Nidhi Gupta
PG Student, ITM University, Gurgaon, India
Sidharth Bhatia
Assistant Professor, ITM University, Gurgaon, India

Abstract:
Driving is a cognitively demanding task. Any distraction or deviation can be potentially dangerous. It is required that any task that is not directly required for driving should not distract the driver.
Gestures have always been an integral part of interaction by humans. Gestures are closely connected to speech. It is rather common to make gestures while speaking. Further, gestures are essential in situations where other communication channels cannot be used. Driving assistance for the deaf, for example.
The system proposed here describes a gesture based method to make the human-computer interaction intuitive and natural. This will do away with the need to have a separate interface (like buttons) which needs learning or conditioning.

Keywords: *Gesture recognition, Computer Vision, Automotive human-computer interface*

1. Introduction

Driving a car is a cognitively demanding task. Any activity (besides driving) that requires the driver's attention can prove to be potentially dangerous. The tasks performed during driving are broadly categorized as primary, secondary and tertiary. Primary tasks are the necessary tasks such as turning the steering wheel, changing gears, pressing the pedals etc. Secondary tasks are the ones that are not directly required for driving, e.g. turning on the parking lights, adjusting the volume of the music system etc. Tertiary tasks are the ones which are not needed while driving, like talking to a co-passenger. [i][ii]

There is a tremendous amount of work done in the area of gesture recognition [iii][iv][v][vi] and providing a natural interface for human computer interaction in a car.[vii][viii][ix][x][xi] The aim of the work done here is to implement the gesture interface for secondary tasks.

In the system proposed here, there is a camera to capture images, a microprocessor to do all the processing and the results will be displayed on a display system.

The camera will capture images and send to the microprocessor. These images will be buffered and stored in a database. The microprocessor will then analyze these sets of captured images to determine if they contain a hand or a portion of it, whether it is similar to any of the pre-defined set of gestures. These pre-defined set of gestures will be stored in the memory of system for comparative analysis. If the images captured correspond to a particular gesture, then the microprocessor will display information about the identified gesture on the display.

The user is expected to make gestures in front of the camera. The system response is via the display system which also shows every step of the processing.

The building blocks used in this system are:

i. ARM based development board: Raspberry Pi Model 2
   A 900MHz quad-core ARM Cortex-A7 CPU, 1GB RAM. It can run full range of ARM GNU/ Linux distributions.
ii. OpenCV (Open Source Computer Vision Library)
   OpenCV is an open source library designed for computational efficiency and with a strong focus on real-time applications. It has several modules which perform specialized functions and are independent of each other.
iii. Logitech C110 webcam

2. Algorithm

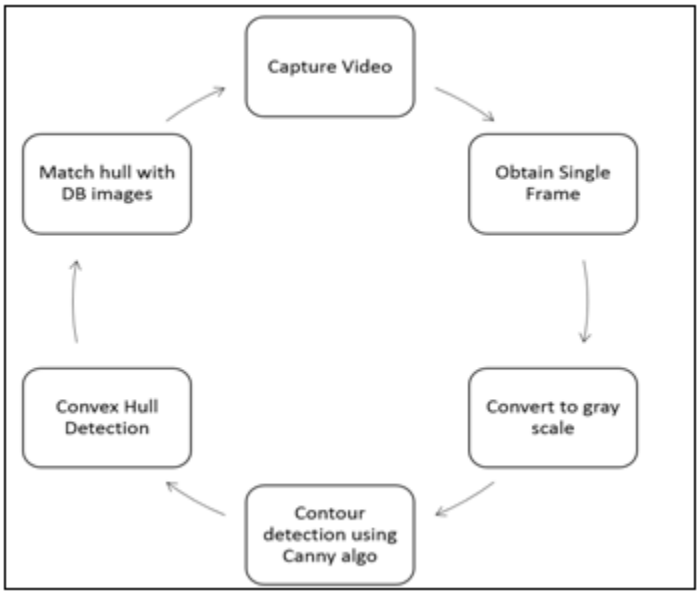

Figure 1: Steps in the algorithm

2.1. Steps

2.1.1. Capture Video

The video sequence of the hand movement is captured using the webcam. The constraint here is that the video sequence must be captured within a time frame such that a fast gesture is also captured. The result should have clear picture of the hand movement rather than blurred or unclear.

2.1.2. Extract Single Frame from the Video

The video sequence captured in the previous step is then analyzed frame by frame. Each frame corresponds to a single image. This image must contain a shape that is similar to the shape of the any of the gestures in the database.

2.1.3. Convert the Frame to Gray Scale

The frame obtained (or a single image) must be converted to grayscale. Grayscale image is an image in which the value of each pixel represents only single sample – intensity information. The image contains various shades of gray, with black being the weakest and white being the strongest intensity. This step is important to perform further operations such as smoothening and edge detection.

2.1.4. Smoothen the Image

Smoothing is a process to create an approximation function that attempts to capture important patterns in the data; leaving out noise or other fine-scale structures or rapid phenomena. The data points of a signal are modified so individual points (presumably because of noise) are reduced, and points that are lower than the adjacent points are increased leading to a smoother signal. The filter used here is Gaussian, which is the most useful filter, though not the fastest. This operation is performed by convolving each point in the input array with a Gaussian kernel and then summing the results to produce the output array.

Gaussian function:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

Here, x is the distance from origin on X-axis and σ is the standard deviation of the Gaussian distribution.

2.1.5. Edge detection using Canny's algorithm

Edge detection is the name for a set of mathematical methods which aim at identifying points in a digital image at which the image brightness changes sharply or has discontinuities. The points at which image brightness changes sharply are typically organized into a set of curved line segments termed edges. The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. Following are the steps of Canny's Edge Detector [xiii]:

i. Gaussian Filter based smoothening in order to remove noise
ii. Intensity gradients calculation of the image
iii. Non-maximum suppression to remove spurious response to edge detection

iv. Double threshold application to determine potential edges
v. Edge tracking: Finalize the edges by removing all other edges that are weak and not connected to the strong edges

2.1.6. Contour Detection

Contour detection is done by the border following algorithm. It performs a topological analysis of digitized binary image (obtained in the previous step of processing). The analysis includes determining the surround-ness relations among the borders of a binary image. Since the hole borders and the outer borders have a one-to-one correspondence to the connected components of 1-pixels and to the holes, respectively, this algorithm yields a representation of a binary image. Features are extracted from this representation of the image, without reconstructing the image. [xiv]

2.1.7. Convex Hull Detection

A convex hull of a set X of points in the Euclidean space is the smallest convex set that contains X. For n number of input points, there are h number of points on the convex hull. The calculation of the convex hull is done using the Sklansky's Algorithm. Sklansky proposed that the structure present in a simple polygon would allow the computation of the convex hull to be carried out in O (n) time. Since sorting is not needed, time complexity would be dominated by the 3-coins loop, which is linear. [xv][xvi] [xvii]
This hull obtained in this step is then used to match the shape of the input image with the one in the database.

3. Block Diagram

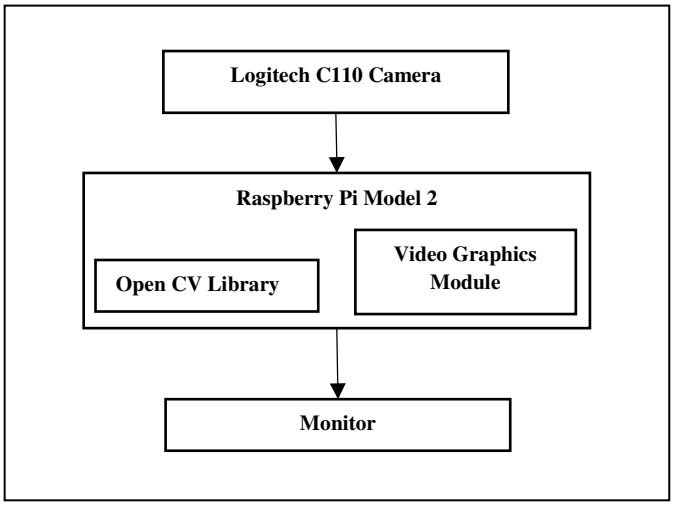

Figure 2: Components

The webcam is used to capture the images. A webcam is used as a standard device to demonstrate that the concept is not restricted to any particular camera. The image sequence is saved and processed in the Raspberry Pi Model 2 development board, using the OpenCV library. The processing of the image sequence is as per the algorithm for gesture recognition. The output of the processing is displayed on the monitor.

4. Advantages & Limitations

This system has the advantage that it is intuitive and easy to use. So, with little learning, the users can learn to use it without having to divert little or any attention. It can be used for secondary tasks while driving, which need little or no distraction.
Limitations of the current system are it is restricted to one user only. That is, the system can recognize only one type of hand. It is not robust yet to recognize hands of varying sizes and shapes.
This can be improved with the use of machine learning. The system should have a feature set about the required gestures. It should be able to recognize those features for different people's hands. And also, train itself on the commonly used gestures and the command that they frequently use.

5. Results

It was observed that the computer vision operations work best on image sequence that has edges which do not touch the external frame. The algorithm implemented here is able to detect gestures that involve counting fingers. It was able to count five fingers accurately. It can be concluded from the above results that this algorithm is accurate for counting fingers and can be used for recognizing such gestures which involve counting. For rest of the gestures, it needs to be adapted.

6. Conclusions

The use of computer vision techniques for gesture recognition is both reliable and efficient. The results obtained from the experiments were consistent.

The current system can be improved to include self-learning module. This will enhance the capability to recognize varied and diverse inputs. Thus, hand gestures of different people (children and adults, skin variations, orientations) will be accounted for.

7. References

i. Sebastian Loehmann , Doris Hausen "Automated Driving: Shifting the Primary Task from the Center to the Periphery of Attention" Workshop on Peripheral Interaction: Shaping the Research and Design Space at CHI 2014

ii. Micah Alpern and Katie Minardo, "Developing a Car Gesture Interface For Use as a Secondary Task", Human-Computer Interaction Institute (HCII), Carnegie Mellon University, 2002

iii. Vishal Nayakwadi, N. B. Pokale, "Natural Hand Gestures Recognition System for Intelligent HCI: A Survey" International Journal of Computer Applications Technology and Research Volume 3– Issue 1, 10 - 19, 2014

iv. Vladimir I. Pavlovic, Rajeev Sharma, Thomas S. Huang, "Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review" IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 19, NO. 7, JULY 1997

v. Alejandro Jaimes and Nicu Sebe, "Multimodal Human Computer Interaction: A Survey" FXPAL Japan, Fuji Xerox Co., Ltd.

vi. David J. Rios-Soria, Satu E. Schaeffer, Sara E. Garza-Villarreal "Hand-gesture recognition using computer-vision techniques" Universidad Autónoma de Nuevo León (UANL)

vii. John K. Lenneman, Joseph Lenneman, Nicholas Cassavaugh & Richard Backs, "DIFFERENTIAL EFFECTS OF FOCAL AND AMBIENT VISUAL PROCESSING DEMANDS ON DRIVING PERFORMANCE" PROCEEDINGS of the Fifth International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design

viii. Sebastian Loehmann, Martin Knobel, Melanie Lamara, and Andreas Butz, "Culturally Independent Gestures for In-Car Interactions" INTERACT 2013, Part III, LNCS 8119, pp. 538–545, 2013

ix. Niall Cairnie, Ian W. Ricketts, Stephen J. McKenna, Gordon McAllister, "Using finger-pointing to operate secondary controls in automobiles" Proceedings of the IEEE Intelligent Vehicles Symposium 2000 Dearborn (MI), USA October 3-5, 2000

x. N. Cairnie, I. W. Ricketts, S. J. McKenna and G. McAllister, "A prototype adaptive finger-pointing interface for operating secondary controls in motor vehicles" IEEE 2000

xi. Martin Zobl, Michael Geiger, Bj¨orn Schuller, Manfred Lang, Gerhard Rigoll "REALTIME SYSTEM FOR HAND GESTURE CONTROLLED OPERATION OF IN-CAR DEVICES" ICME 2003, IEEE

xii. Pedro Trindade, Jorge Lobo and Joao P. Barreto "Hand gesture recognition using color and depth images enhanced with hand angular pose data" IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI) September 13-15, 2012. Hamburg, Germany

xiii. Canny, John. "A computational approach to edge detection." Pattern Analysis and Machine Intelligence, IEEE Transactions on 6 (1986): 679-698.

xiv. Suzuki, Satoshi. "Topological structural analysis of digitized binary images by border following." Computer Vision, Graphics, and Image Processing 30.1 (1985): 32-46.

xv. Homma, Kazuhiro, and Ei-ichi Takenaka. "An image processing method for feature extraction of space-occupying lesions." Journal of nuclear medicine: official publication, Society of Nuclear Medicine 26.12 (1985): 1472-1477.

xvi. Sklansky, Jack. "Finding the convex hull of a simple polygon." Pattern Recognition Letters 1.2 (1982): 79-83.

xvii. Sklansky, Jack. "Measuring concavity on a rectangular mosaic." IEEE Transactions on Computers 21.12 (1972): 1355-1364.