Assistive Device for Hand Amputees
-
Documentation

Group 07

Shaswat Babhulgaonkar-115011103
Pritam Bendkule- 115011204
Jayesh Suryavanshi- 115011184
Vivek Sanap- 115011307

Abstract

In this article we describe ADHA - Assistive Device for Hand Amputees. We describe its features, use-cases and finally give technical details about its usage and implementation

1 Introduction

ADHA is a device that is designed to support people who have had both their arms amputated above or below the elbow. It provides two major types of support:

1. Enable the user to connect to the digital world by forming a convenient interface to a computer's keyboard and mouse.
2. Enable the user to transport heavy objects by having a transport robot that they can control

These two target areas were chosen with some care. This system was designed keeping mind the needs and abilities of one arm amputee who's arms were cut off just below the elbow.

In the modern world, it is essential to be able to connect to the massive amount of digital content available to us. Yet amputees have trouble operating the commonly used Human-Computer Interfaces and are hence denied access to the digital world. We aim to remedy this situation by building into ADHA the ability to enable arm amputees to interface with personal computers. The interface is one that amputees can use naturally and fits in well with the physical resources at their disposal.

The lack of a full hand also means that arm amputees often have difficulty holding heavy objects for *extended periods of time*. Hence we have set out to create a robot that can serve as a **portable table** for the users so that they can transport and access objects with greater mobility.

2 Overall Structure

For using the keyboard, the user can rely on the Operating System's speech recognition software. However speech recognition as of 2015 is not very robust and it can become frustrating for the user if the speech recognizer makes a lot of mistakes. To remedy this, ADHA allows the user to make keypresses. The encoding scheme is designed to make the common keys easier to 'press'.

The mouse controller is very robust and easy to use. In trials, the users were able to easily control the computer mouse. This is very essential especially because speech recognition does not support operations that are to be performed using the mouse very well.

One of ADHA's central elements is a *hand gesture recognition system* that works by having a motion sensor attached to one of the person's elbows. It also sports a button that the user can press with the other arm. When the user presses the button, ADHA reads from the sensor. With this the user can communicate one of six signals to the system. This is done by orienting their arm in on of six possible directions, one along each of the x, y and x axis in each direction, which determines which way is 'down' for the sensor. The sensor uses an accelerometer to sense which way is down and converts this reading into one of 6 possible numbers from 1 to 6. For the rest of the system (except the mouse controller), this number is all that is ever used. The rest of the system has an encoding scheme with which the user can communicate with the system. This is further described in Section 2.

ADHA's robot is a prototype of a strong mobile 'table' that can be controlled by the user's gestures. The user can command this robot to go for-

ward, backward, turn left and turn right. This is through intuitive hand gestures in the robot mode. Currently the robot is only for mobility, in the future it would be worthwhile to implement one with a robotic arm. This could serve two purposes. One, it will allow the user to pick things up without having to pick it up themselves. Two, it can also serve as a replacement to their actual arm. Note that this system was designed keeping in mind the needs of one amputee, known to one of the team members. This person was strong enough to lift objects and in general life life almost as if everything were normal. However he/she would tire of lifting weights for extended periods of time. Hence this robot would considerably ease the life of such a person.

3 The Hardware

ADHA consists of three major parts. A pictorial representation can be found in the accompanying videos. Below we describe the major components:

The Microcontroller

The microcontroller hangs from the user's neck like a pendant, in a style similar to that of Sixth Sense. It is powered using a portable charger of the kind that is commonly available. To lighten the weight, this can be kept in the user's pocket. In the prototype we use an Arduino™ Mega™ board. However naturally in production a much lighter and cheaper microcontroller will be used.

Indicator Lights

There are four indicator LEDs on the system. Three of them are used to indicate which mode the system is currently in. The fourth is lit up when the user presses the button as feedback.

The Motion Sensor

The motion sensor is a Brigosha sensor card which has three ICs that contain an accelerometer, a gyroscope, a magnetometer, a pressure sensor and a temperature sensor. It is mounted onto the user's elbow using an elbow

band. Wires go from the sensor to the arduino pendant over the user's shoulder

The Button

The button is essentially triggered when two plates come in contact with each other. One plate is attached to the user's side and the other to the user's other arm. When the plates come in contact, a circuit is completed which pulls one of the arduino's pins to VCC. When not in contact, the pin is grounded through a 480K resistor. The large resistance ensures that the VCC and ground are not short circuited when the user completes the circuit.

The Robot

The robot prototype is a four wheen differential drive machine that is controlled by the XBee wireless module from the user's arduino pendant. The robot acts as a mobile table for the user, enabling them to conveniently transport bulky objects. Currently the robot does not carry much weight due to budgetary limitations, however the prototype successfully demonstrates the potential efficacy of such a system.

4 The Modes

The Select Mode

This is a mode to select other modes. The user can 'press' four of the six signals to transfer to one of the other four modes. Further, from each of the four other modes, we have defined a way to come back to this mode. Also, the system starts of from this mode. So whenever the user wants to change modes, the user first comes back to this mode and then goes into one of the other modes.

Common Keys Mode

We recognized that most of the time we are using only a very small number of keys. So we have strived to make the frequently used keys as accesible as possible. Hence this mode supports five keys, each of which can be pressed using just a single gesture. Currently we have programmed it to support:

- Backspace - to correct any errors made by the speech recognition software
- Up and Down Arrow keys - to enable easy scrolling, among other things
- Enter Key - Very commonly used
- Alt + Tab - Very useful for switching windows
- Go back to select mode so that the user can switch modes

Full Keyboard Mode

This implements all the keys in the keyboard. To make the user's life easier we use **variable length encoding**. That is, the commonly used keys need two hand gestures and the uncommon ones require three. Note that we can represent two hand gestures using a number between 1 and 36 and we can represent three hand gestures using a number between 1 and 216. If the number is less than 21, it is interpreted as a symbol requiring two hand gestures, else the system expects a third hand gesture. Pressing three 1's makes the system move back to the select mode

Mouse Mode

In this mode, the user controls the mouse using the position of their arm. The mouse is stationary when their arm is vertically downward. Beyond that to moves in the direction at which the user point, the speed depending on how much the user tilts their arm. We recognize that the user would want a some freedom in keeping the mouse at rest. So we scale the magnitude of motion between 0 and 1 and then **cube** it (do that the derivative is 0 close to the center position). This ensures that there is a large region of slow mouse movement and the mouse really only moves by large amounts when the user wants it to. Pointing the arm above the 'horizon' exits the mouse mode and puts the system in the select mode.

Robot Mode

This mode controls the robot. The user can make 6 gestures. Four of these encode forward, backward, left and right motion. That is, when the user makes that particular gesture the robot moves in the designated direction.

The fifth gesture puts the system back into select mode. Note that the robot is a differential drive bot. This means that left and right means a zero degree turn toward left and right. The amount of turn depends on the duration of the gesture (ie. the duration for which the button is pressed)

5 Robust Gesture Recognition

Gesture recognition is implemented using the accelerometer in the sensor card. First the accelerometer data is put through an exponentially weighted moving average. This gets rid of the large amount of accelerometer high frequency noise. We find that this system performs better than when the accelerometer is coupled with the gyroscope using the complementary filter. We believe this is because of some calibration issues in the gyroscope. There are two primary modes of reading the accelerometer data

1. Descrete Mode: In this mode, we simply pick the most dominant component of the acceleration as use that as our symbol. There can be 6 dominant directions when projected along cartesian coordinates. Hence our system supports six symbols per gesture. The symbols are very widely spaced and hence can be very robustly recognised. This is suitable for an amputee who may not always be able to make very precise gestures. The readings are taken when the button is pressed.

2. Contiuous Mouse Mode: This is used for the mouse. Here we essentially use the angle of inclination of the arm with respect to the vertical downwards direction as a measure of where to move the mouse. Further we normalize this value to between -1 and 1 and then cube it to give a 0 derivative at vertical position. THis is so that the user can be comfortable while the mouse is not supposed to be moving