Automotive Gesture Recognition Design Document

May 13-07
Scott Schroeder
Ethan Little
Thomas Moser

Version	Date	Author	Change
0.1	10/15/12	TM	Initial Document
0.9	10/24/12	TM,SS,EL	Section 2 updated

Introduction

1.1 Purpose

The purpose of this document is to outline the technical design of the Garmin Vehicle Gesture Recognition System and provide an overview of the systems implementation.

This document will:

- Detail the functionality which will be provided by each component or group of components and show how the various components interact in the design
- Provide a basis for the gesture recognition system's detailed design and development

1.2 Scope

The gesture recognition system design outlined in this document builds upon the scope defined in the Requirements phase. The project encompasses all research and hardware testing as well as implementing gesture recognition algorithms. A final product will not be delivered beyond a proof of concept and details of all algorithms and designs used.

1.3 Definitions, Acronyms, Abbreviations

Term	Description
IR Sensor	Infrared Sensor
CMOS	complementary metal-oxide-semiconductor (integrated circuits)
USB	Universal Serial Bus

1.4 Design Goals

Reliability

The ability to recognize gestures in an automobile setting with an accuracy of 99%. The project should have almost no false positives and failure should be modularized and not affect other systems. As a vehicle application product, all systems need to perform to a high degree in extreme temperatures and lighting conditions.

Response Time

The time from a user action to the corresponding results should be minimal. The webapp should not cause the user to wait more than a few seconds for results. All image processing should be done with almost real time results. The detection of gestures is allowed a longer processing time period but should be reasonable.

Scalability

The system will be designed to recognize a small set of gestures but should easily be updated to recognize 15+ distinct gestures. All software should be portable to varying powered hardware with minimal changes.

Standards

Numerous standards should be followed in the development of the project. The automotive industry has safety and interaction standards that should be complied with as well as various electronics standards. The standards-compliance will apply to design, platform, and application development. Examples of standards include C/C++ and USB.

Cost

All purchasing will be approved by both the project advisor and Garmin. Hardware to be requisitioned includes:

- CMOS camera - $10-$30
- Microcontroller - $50-$150

A final cost of less than $200 should apply to this project.

Expected Deliverables

A final producible product is not expected or planned but detailed designs and algorithms as well as software may be delivered. The goal of this project is to prove and display gesture recognition on a microcontroller in an automobile.

Specifications

2.1 Input/Output

The system will use IR and CMOS sensors to capture gestures from the user in the center of the car's cockpit, in front of the infotainment system. Output will be a display on a monitor showing what gesture was recognized.

2.2 User Interface

The interface provided to the user will consist of a simple GUI displayed on a monitor that shows an indication of what gesture was recognized. After starting of the program on the pc all interaction will be done using gestures in front of the camera/ir sensor.

2.3 Hardware

Sensors

There were many types of sensors under consideration for receiving input from the environment. Foremost among these were sensors relying on three different technologies.

1) Ultrasonic Array - Use a number of sensors that emit high frequency sound waves and then receive the reflection and use the time of propagation to calculate depth.

Pros:

a. Works well with small number of gestures.

b. Inputs smaller than other technologies making processing more efficient.

Cons:

a. Only effective for a small number of gestures (~4-6)

b. Relies on sound, decreasing response time compared to other technologies.

2) CMOS Camera - Use a simple CMOS camera to gather input images and then run image processing algorithms to determine hand orientation and shape for each frame. Then run further algorithms to detect gestures over time.

Pros:

a. Allows for a large number of gestures (15+).

b. Based on light input which is much faster than other technologies(e.g. Ultrasonic).

Cons:

a. Hard to filter out background noise.

b. Issues handling different skin colors (Most algorithms filter out non skin colored pixels as background).

c. Large inputs requiring heavy cpu usage.

3) Infrared Depth Sensing Array - Two dimensional grid of infrared beams are projected on a surface. An infrared sensor watches the grid and takes in images. Depth is determined by the spacing of the beams and returned as a depth grid for processing.

Pros:

a. Background noise easily filtered using depth

b. Lighting in cabin doesn't affect usability.

c. Allows for large number of gestures (15+).

Cons:

a. Hardware varies and needs to meet space criteria.

b. Requires advanced drivers and software

Chosen Solution:

The infrared Depth Sensing Array was chosen to be the final input device. This decision reflects its ability to allow for a large number of gestures and the fact that the outputted data is easy to manipulate. Two of the main concerns when looking at the sensors was the ease at which background noise can be filtered out and the ability to function regardless of operating conditions (e.g. variable lighting).

Main Platform

Initial development of the system will be completed on a pc running a variation of Linux. This will allow development to occur without initial hardware constraints and ensure algorithms are emphasized more than hardware interfaces. After a fully functioning system is finalized the goal will be shifted to modifying the software to run on a scaled down Linux distribution running on a microcontroller.

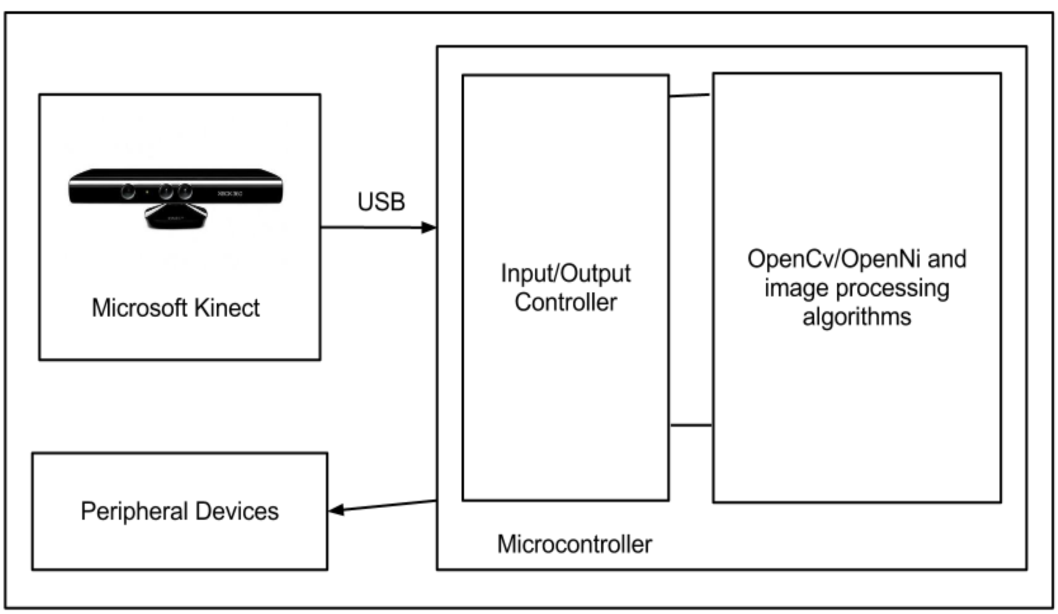


Fig. 1 Gesture Recognition Block Diagram

2.4 Software

OpenNi

OpenNi is an open source framework used for writing applications that utilize natural interaction such as video and audio input. OpenNi provides an API that facilitates communication with both low level and high level devices (e.g. Kinect).

OpenCV

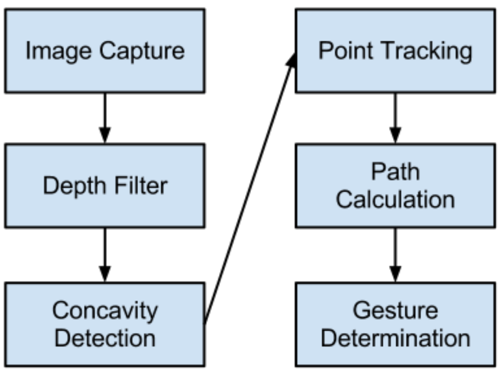
OpenCV is an open source framework similar to OpenNi that works to increase the possibilities of computer vision as well as provide an easy to use interface between both low level and high level devices. For the purposes of this project OpenNi and OpenCV will be used together to provide functionality for the handling of data from the Kinect sensor.

Operating System

For initial development a computer running a full version of linux, preferably Ubuntu 10.10, will be used to program and test the gesture recognition algorithms. After the algorithms have been tested for accuracy and efficiency the program will be ported to a smaller board (e.g Rasperry Pi, Gumstix, or BeagleBoard) that runs a trimmed down version of Linux.

Process

The following process shall be applied when processing gesture input from the sensor.



2.5 Testing

Testing will consist of unit tests for each of the individual software modules as well as in-depth hands on testing of the system. The hands on portion of the testing will consist of volunteers coming in and making gestures in front of the sensor. The first portion of the volunteer testing will be spent having them make gestures from our defined list of gestures. The second portion will consist of the volunteers acting like they are driving normally and using their normal in car gestures. This will help determine the false-positive and true-negative rates of our system so that we can go in and fix our algorithms to handle these cases.