# SYSTEM REQUIREMENTS SPECIFICATION

## CS308 PROJECT

**17 FEB 2012**

## SLAVE BOT

**GROUP 04**

**VARUN REDDY K -- TL**

**RAVI VOODA**

**TARUN GUJJULA**

**RAJESH KAMINENI**

**Submitted in partial fulfillment**

**of the requirements of**

**CS308 Embedded Systems**

## 1. Introduction:

The purpose of this document is to present a detailed description of the Gesture Controlled Slave Robot. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for the developers of the system.

### 1.1 Definitions, Acronyms, Abbrevations

| Term | Definition |
|---|---|
| Bot / Robot | The electro-mechanical machine that is guided by the user. |
| Kinect | The motion sensing input device developed by Microsoft. |
| Software Requirements Specification | A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. |
| User | The person that is currently controlling the bot. |
| Natural User Interface (NUI) | An evolving model for human-computer interaction that is context-appropriate and adaptive. Because the NUI exploits a user’s existing skills and expectations, it is easy to learn. A NUI might incorporate speech, gesture, touch, or location. |
| Depth data | The data that is produced by the depth image camera in the Kinect sensor. Each frame in the stream contains the distance, in millimeters, to the nearest object at a particular x and y coordinate in the depth sensor's field of view. |
| Skeletal data | The data is provided to application code as a set of points, called skeleton positions, those compose a skeleton. This skeleton represents a user’s current position and pose. |
| VGA | Video Graphic Accelerator |
| SDK | Software Development Kit |
| IDE | Integrated Development Environment |
| IR | Infra-Red |
| API | Application Programming Interfaces |
| DirectX | DirectX is a collection of API for handling multimedia related tasks, especially game programming, on Microsoft platforms. |
| Visual Studios | IDE for using the SDK used. |
| AVR Studios | IDE for c programming for the robot. |
| X-CTU | Software used to configure ZigBee wireless module. |

### 1.2 References

- Programming Guide : Getting Started with Kinect for Windows SDK Beta
- Code Walkthroughs
- Kinect SDK Quick Starts
- Programming Guide : Getting Started with Visual Studios

## 2 Overall Description

### 2.1 System Environment

**Figure 1 - System Environment**

The Gesture Controlled Robot has two active systems and one coordinating system. The communication between the coordinator and the robot is on a dedicated wireless frequency channel. The User is able to view the problem if any are reported by the bot on the coordinator.

### 2.2 Product Perspective

The robot (Firebird V) is controlled remotely using the ZigBee wireless interface. The user is providing the input by movements representing gestures that are captured by the Kinect sensor device. The gestures captured by the Kinect sensor are translated to a particular command using the Kinect API and to signals using Esterel for the robot. Kind of gestures we are trying to implement are, on pointing our finger to a object kinect will send the location of the object to robot. Bot will go and collect the object and place it in trash or at some predefined location.

## 2.3 Product Functions

This section outlines the use cases for a user in this system.

### 2.3.1 Robot Movement Case

#### 2.3.1.1 Use case: Robot Movement

Diagram**:**

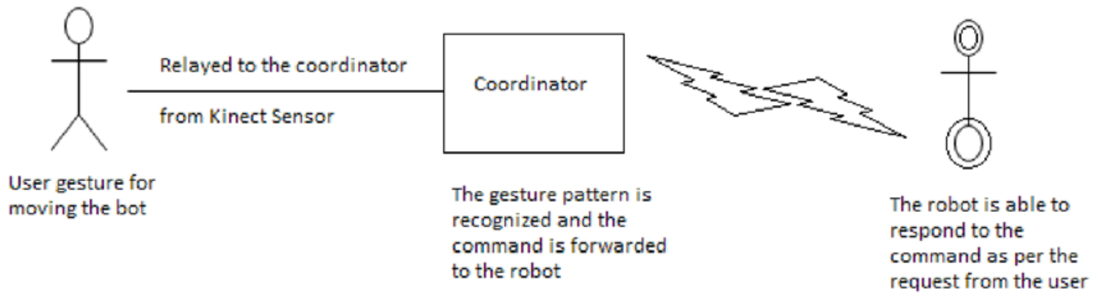
Relayed to the coordinator
from Kinect Sensor
Coordinator
User gesture for moving the bot
The gesture pattern is recognized and the command is forwarded to the robot
The robot is able to respond to the command as per the request from the user

**Figure 2 – Robot Control Process**

**Description:**

The User moves in the pattern that is recognized as a gesture by the Coordinator. The command corresponding to the particular gesture is forwarded to the robot. There can be many possible movements for a robot. Our robot can perform actions like moving to a specified point, lifting a object.

**Initial Step-By-Step Description**

Before this use case can be initiated, the user must be recognized by the Kinect and the wireless communications must be active.

1. The User chooses the movement needed to be performed by the Robot.
2. The User must perform the appropriate gesture.

3. The gesture is recognized and translated into a command by the coordinator and the command is forwarded by the robot on the wireless.
4. The robot accesses the command and fires a signal for movement.
5. The robot perform action as per the signal generated.

## 2.4 User Characteristics

The User is expected to be able to perform the gestures fairly consistently and with a good level of resemblance so that it can be detected properly by the Pattern Recognizer on the Coordinator. The user is expected to stand in fixed location while pointing his finger.

## 2.5 Constraints

- The Kinect SDK is restricted to work on MS Windows 7 with DirectX SDK; same will be the restriction for the software developed.
- The Kinect Sensor Device requires a good processor and a dedicated VGA card; same will be the requirements of the machine used as the coordinator.
- As the bot moves relative to its location, the user must point to a location which is in the capturing field of the kinect.
- The bot must not encounter any obstacles in the location it is guided to move.
- The bot will be commanded to lift only paper weights.

## 2.6 Assumptions and Dependencies

- The bandwidth of ZigBee Communication should be able to handle the load of communications i.e. downlink **-** the commands from the PC to the bot; and uplink **-** the video from the camera mounted on the bot to PC + Errors if any.
- There must be no form of interference between the user and the Kinect Sensor as this will affect the tracking mechanism of the sensor and may lead to the sensor resetting itself.

### 2.7 Requirement Subsets

- Gesture Recognition
- Transmission of Commands
- Interpretation of Commands
- Signal generation based on Commands
- Action performed based the Signals
- Response based on the environment of the Robot

# 3 Details

## 3.1 Functionality

This is a Remote Robot Controller System for a user. This system will be designed to maximize the user's productivity by providing an interface to control the robot in a most natural way possible. By removing any hand holding device to control the robot the user's ability to control the bot is improved as the interface is now hands free, more analog. The usability to the system is improved without compromising the ease of understanding the same.

More specifically, this system is designed to allow a user to control and communicate with the bot via remote wireless connectivity. The wireless network is configured to provide point to point connectivity between the coordinator and the robot ensuring no external interference on the communication channel. The software will facilitate recognition of the movement patterns of the user to be detected as gestures which in turn are converted to signals used to control the Robot. A set of preformatted responses from the bot can be used to detect any problems faced by the bot to perform the action or provide information on the environment. Also, a camera mounted on the bot so that the user can see the environment of the bot as live video rather than written information (which can also be included).

## 3.2 Supportability

### 3.2.1 Basis for future work

This will form the basis of more sophisticated projects using Kinect in the coming years.

### 3.2.2 Distinguished Modules of the Project

There are three distinguished modules of the project:

1. Kinect programming using Microsoft APIs for Natural User Interface for gesture recognition.
2. ZigBee communication module for wireless communication.
3. Esterel signal generation for different commands.

Any of these modules can potentially be used in other projects requiring their functionality.

### 3.2.3 Documentation

Documentation in terms of embedded comments and other explicit write-ups will help others in future.

## 3.3 Design Constraints

The restrictions being implemented in the code are primarily due to those imposed of the system by the hardware being used. They are as follows:

- The ZigBee Wireless Communication Module uses a Serial COM Ports. So, all the communication between the coordinator and the Robot will be via a serial port only.
- Microsoft Kinect SDK and DirectX SDK require Microsoft Visual Studios to be used to allow proper access of their libraries. So, the use of such SDKs will require Visual Studios explicitly.
- In order to have a 360 degree view of the robot, the user will have to turn the robot 360 degree rather than turning the camera because of the absence of motor mount for the camera (this is an option chosen).

## 3.4 Interfaces

### 3.4.1 User Interfaces

This is the part of the system that will be used to interact with the user i.e. recognizing their movements as well as displaying information for them to be used for bot navigation. They are as follows:

- Kinect interface view of the user which includes the skeletal view, the camera view and the IR greyscale distance measurement view of the user.
- A display showing that the recognized gesture.
- A display showing the camera view of the robot and the environment report from the environment.

### 3.4.3 Hardware Interfaces

This section states the essential physical equipment required to interact with the application designed. They are as follows:

- Kinect Sensor Module to detect the movements of the user.
- Computer System with the following essentials are Kinect Sensor Module to run:
  - Intel Core2Duo 2.66GHz
  - Dedicated VGA card with DirectX Support
- A camera to be mounted on the robot with the data bus connectivity to the ZigBee communication module.

### 3.4.4 Software Interfaces

This section states the requirements to development the application which will use the data feed from the various interfaces and use them to control the bot or display information for the user. They are as follows:

- Kinect SDK which in turn requires the following:
  - Windows 7
  - DirectX SDK
- Visual Studios
- Atmel AVR Studios

### 3.4.4 Communications Interfaces

This section states the requirements to create a communication between the robot the coordinator. They are as follows:

- 2 ZigBee Wireless Modules
- X-CTU ZigBee Configuration

## 4 Quality Control

### 1.1 Recognizing Various Hand Gestures

A particular *type* of gesture will correspond to a particular command. All possible instances of that type of gesture should be manifested as that command.

*Example:* If system defines ‘clap’ as a gesture corresponding to command ‘stop’ then all possible gestures that represent clapping should stop the bot.

### 1.2 Sensing the Object

The Bot uses the hand gesture input to locate its destiny. After reaching the point, it uses IR-Range sensor to sense the object in the near environment. This functionality hence improvises for better performance.

### 1.3 Buffering

While performing a task if the robot is given new destination, the robot implements the buffering technology in order to save its destination’s co-ordinates and hence promoting the ability of the bot.

## 5 Risk Management

1. Gesture Recognition: A single pattern made by the user will never be an exact match to itself if tried repeatedly. So, the recognition program will have to be sufficiently lenient so as to allow some liberties to the user but not to misinterpret a general movement for a gesture. So, in order to confirm a gesture, a trial run will be needed prior to using the bot to confirm the movement pattern is recognizable as a gesture during the trial runs.

   · Fall Back Plan: The gestures are pre-programmed in the code.

2. Gesture to Signal Transition: The body movements of the people are made in the real world, so they will be in three dimensional and will be continuous. We will be required to analyze the input data for changes in body points during pattern formation and relay the information to the bot at the same time without any break in command flow. This will ensure that the bot is stationary in absence of signal from the user.

   · Fall Back Plan: A signal to perform a previous gesture is emitted till a new gesture is recognized.