# Project Report for Group 16

# Gesture Controlled Robot

## Introduction

As the world moves towards providing a Natural User Interface for day-to-day applications, we try to use the same in the field of robotics. With the release of the Kinect SDK by Microsoft Research, many possibilities have opened up which are yet unexplored. The SDK, still in its nascent stage, is yet to be experimented with to the fullest. We try to step onto this untried road in the journey towards a plethora of gesture controlled devices.

## Problem Statement

Make a robot navigate based on the gestures of the user. Provide the user with a view around the robot.

## Requirements

* 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

## Implementation

### Gesture Recognition

To recognize the user’s gestures, we used the Microsoft XBOX 360 Kinect(C). We interfaced the Kinect with a laptop using the Kinect SDK provided by Microsoft. We developed the system using the Visual Studio IDE.

The Kinect gathers the color and depth information of the using an RGB and Infra-Red camera respectively. The Kinect SDK then uses this data to recognize a human blob. It then creates an approximate skeleton of all the limbs of the blob detected. From this, the Kinect can make out the location of the various joints(hands, neck, head, etc.) of the human body.

The SDK then provides this information about the joints in an event that is fired regularly based on the frame-rate of the transmission of data from the camera. Using the position of joints from the current as well as previous frames, we can interpret the movement/position of the hand as various gestures.

Initially, we targeted static gestures, i.e., gestures based on the position of, say, the hands. This gesture can be interpreted based on the data from one frame only. Later we moved towards deciphering more complex movement-based gestures like waves and claps. This was done using the data collected over time.

### Transmission of Commands

Since the robot may be at some distance from the user, we used wireless communication to control the robot. This was done using a Zigbee network. A Zigbee NIC was attached to the laptop and another was attached to the robot. This enabled us to control the robot even from some distance.

### Interpretation of Commands

The commands were then interpreted and then the robot was controlled accordingly.

### Giving the user a view of the environment of the robot

A wireless camera was mounted on the robot. The receiver was interfaced to the laptop and the video would give the user of the surrounding of the robot.

## Testing Strategy and Data

The testing consisted of various gestures. The strategy was to detect the accuracy of the detection of the gestures. The static gestures were easy to test as they merely consisted of positioning the hands in the appropriate position. The motion gestures were a bit complicated as they weren’t so easily detectable by our algorithm. The motion gestures need to be replicated as accurately as possible.

The test data consisted of positioning the hands in various places based on the position of the head. The motion gestures consisted of moving the arms according to the gestures, albeit in a manner that closely resembled the actual gesture.

The static gestures were detected accurately. The motion gestures, however, were not detected so accurately.

## Discussion of System

The main breaking ground was setting everything up and getting a simple sample to work. The system has been set up in a manner so that the installation need not be done again. A basic idea has also been given so as to how the gestures can be detected. The ideas can be explored further. The communication module was also set up successfully in Visual C# and worked seamlessly.

The idea of using these gestures to generate signals was also explored. We figured out as to how this can be done. In the script provided for compilation (buildHash.pl), a signal needs to be added and a function corresponding to this signal needs to be added in the WinAVR.h header file provided. The function needs to receive the command sent by the computer via the Zigbee interface. The robot’s controlling program written in Esterel should wait on these signals and carry out the actions appropriately. This idea however could not be implemented due to lack of time.

## Future Work

This has opened a whole new direction in terms of Natural User Interface for Robotics. As we had mentioned earlier, the sky is the lower limit. The robot can be used to replicate human actions in scenarios where it is infeasible for human beings to be present. An example of such a scenario would be diffusing bombs.

This work can also be extended to game-playing robots where the robot can be used to carry out actions in response to the user’s actions.

Speech also can be interfaced as the Kinect has a microphone facility with echo cancellation. The Microsoft Speech SDK can be used for the same.

## Conclusions

The experiment of Natural User Interface has opened up large avenues for robotics. It adds a new dimension to controlling the robot where the user can carry out actions naturally to him rather than having to train himself according to the robot. By enabling the user to speak in a high-level language of gestures, it has allowed him/her to focus more on the task at hand.

The proof of the feasibility of such a solution in such a short amount of time has also opened up the possibility of various new robots in areas previously unthought-of before.

## References

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