Course Project Documentation

CS-308 Project

**Kinecsterel**

Team: 1

Priyank Parikh 09005006

Nitant Vaidya 09005007

Lalit Swami 09005028

Smit Patel 09005002

**Table of Contents:**

1. Problem Statement 3

1. Requirements 4
2. Implementation 5
3. Testing 9
4. Discussion of System 10
5. Future Work 12
6. Conclusion 13
7. References 14
8. **Problem Statement**

In this project we aim to establish a Kinect-Esterel interface to control bot motions using esterel signals directed by hand gestures.

An operator stands before a Kinect device and makes a hand gesture (which has to be amongst a selected set of hand gestures). The hand gesture made is identified using the Kinect device and its gesture code is transmitted wirelessly over to the bot. Esterel code on the bot takes this gesture code as input and generates appropriate output signals.

With this, we enable two bots to play football with each other, being controlled by a human player each.

1. **Requirements:**

**Hardware Requirements:**

1. One Microsoft Kinect Device
2. Two Firebird Bots
3. Two ZigBee modules

**Software Requirements:**

1. Microsoft C# Visual Studio 2010
2. Microsoft Kinect SDK
3. Perl Interpreter
4. Esterel related files – firebird\_gen, firebird\_winavr.h, buildhash.pl
5. AVR Studio
6. Bootloader
7. X-CTU
8. **Implementation:**
9. **Gesture Recognition**

The C# Project named GestureCapture takes input from the Kinect device and decides if the operator has made any of the gestures. There is a mapping of gesture to gesture code, and the corresponding gesture code is sent to the device.

**Forward**

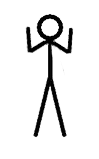
**Right**



**Left**



**Reverse**



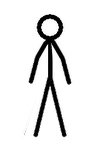
**Hard Right**

**Hard Left**

**Stop**

1. **Transmission of gesture code**

We use serial port communication over ZigBee module to achieve this.

1. **Addition of ZigBee as an input signal in Esterel**

In order to do this we added “ZigBee” in the list of input signals. We added a global variable called ‘gesture’. We added a function which we linked to the input signal “ZigBee” which simply returned the value of this global variable. We also wrote a signal processing routine which was called whenever a new gesture was received. This routine would take the new gesture code and update the value of the ‘gesture’ global variable to that.

Adding of the signal was added to the “buildhash.pl” file.

Adding of signal processing routine was added to the “firebird\_winavr.h”

1. **Defining Esterel output signals based on ZigBee input**

Here, we simply look at the value of the gesture code and set the appropriate output signals. This code snippet below shows how this is done

//for forward motion

If gesture == 1

emit MOVE\_FWD

emit LEFT\_MOTOR\_SPEED(100)

emit RIGHT \_MOTOR\_SPEED(100)

1. **Testing**

The different modules of the project were independently tested before communication was established and the final testing was done

Gesture recognition was first verified by printing out the gesture name and gesture code as the operator made the gestures

Esterel code was debugged and tested by sending signals using X-CTU. We also tried many sending signals in quick succession to see if the bot responded quickly enough, and we found that it did

Testing of the combined modules was done using different people as operators whose gestures are to be captured. We also did two player testing, and checked if one’s gestures are interfering with another’s bot, and we found all of the above criteria to our satisfaction

**Performance Metrics**

Delay: This was found to be really low, response time of the gesture recognition code as well as the gesture transmission time was found to be negligible and hence the bot was found to be very receptive to the incoming gesture

Degrees of freedom:As of now, the Firebird can have 7 states of motion.   
Forward, Backward, Hard Right, Hard Left, Right, Left, and Stop with the possibility of an easy increase to more states in the future.

1. **Discussion of System**
2. **What worked as per plan?**
3. Kinect Gesture Capturing:

The Kinect Gesture Capturing has been completed exactly as planned in our original SRS, with well-defined unique gestures captured and mapped to different bot motion-control functions

1. Esterel Input Signal Addition:

An input signal has been added to Esterel successfully so that the Esterel code on the bot can take input via through Zigbee from the Kinect device connected to the computer

1. Zigbee Communication:

The Zigbee transmission from the computer to the bot, along with configuration of the Zigbee chip is as planned

1. Bot Motion:

The bot motion is as planned in the project. There is no delay between gestures and the motion of the bot and each of the states of motion are executed as intended

1. **What we added more than discussed in SRS?**
2. Two Player Support:

In the original SRS the plan was to control one bot using a Kinect. However for demonstration purposes we added support for two players standing in front of the same Kinect device independently controlling their own bot with hand gestures. Thus we enable two persons to play football against each other with their bots.

1. Auto Switching between Player modes:

Besides allowing for two player control, we switch between one player and two players automatically.

The way we have done this is by detecting the number of persons standing in front of the Kinect. If one person is detected, he/she is selected as the first and default player 1. If 2 persons are detected then their gestures are read separately. These readings do not interfere with each other.

1. **Changes made in plan**
2. Not using C# Zigbee library:

The original plan was to import a library for Zigbee In C# and then use that to transmit from the Kinect C# code to the bot wirelessly. However this failed to work because of some port communication issues which we were unable to iron out.

Instead, we used Serial Port communication in C# and created a port with the specifications set to the Zigbee module

1. **Future Work**

Better Control

* Currently there are 7 gestures : Right, Left, Hard Right, Hard Left and other motions.
* There is scope to increase the detail of this motion and make it move left or right by a certain degrees based on the angle of tilt of the hand. This allows for much better bot control
* This would be more useful from a football playing perspective since a high measure of control is desired

Using Zigbee Channels

* Currently we are using two different Zigbee chips to transmit independently to two different bots
* A possible enhancement is to use different channels on single chip to transmit to two bots
* This might cause a little delay but that would be negligible. It would reduce hardware requirements and also the need to configure the extra zigbee module

1. **Conclusion**

Our code has achieved its purpose of controlling a bot using hand gestures captured by the Kinect. Since the code has been written in a modular fashion, a high level of abstraction is achieved.

This abstraction is on two levels.

Esterel – Zigbee : Owing to the addition of ZigBee support in Esterel by including an input signal, any wireless input can be used by esterel code as input on the bot and appropriate esterel output signals can be generated as per requirement

Kinect Zigbee : The gesture recognition code can be used to transmit wirelessly to any other bot and any other system which uses Zigbee for input. Esterel code for an entirely different architecture can be written which uses the exact same gesture recognition code for input

Hence modularity led to a high level of abstraction and future usability

**Examples**

* A chemical detector bot which receives input from various sensors of chemical levels in a factory or LPG levels in a house and wirelessly transmits this data to a bot. This bot can be coded in Esterel and take wireless input via Zigbee
* A mechanical arm which is controlled by Kinect gestures and coded using Esterel. It can mimic hand actions allowing for better control and be used in locations inaccessible to the human hand

1. **References**
2. Kinect Skeleton Tracking Sample Code: <http://channel9.msdn.com/Series/KinectSDKQuickstarts/Skeletal-Tracking-Fundamentals>
3. Esterel Manual: <http://www.cse.iitb.ac.in/~cs684/esterel/Programming_Firebird_Esterel_manual.pdf>
4. Zigbee Manual:  
   <http://www.ladyada.net/make/xbee/configure.html>

**Programs Installed**

1. Microsoft Visual Studio Express 2010:

<http://www.microsoft.com/visualstudio/en-us/products/2010-editions/visual-csharp-express>

1. Microsoft Kinect SDK: <http://www.microsoft.com/download/en/details.aspx?id=28782>
2. Putty  
   <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>
3. FileZilla  
   <http://filezilla-project.org/download.php>